GRAN ENCICLOPEDIA INFORMATICA

INFORMATICA BASICA/3

EDICIONES NUEVA LENTE

GRAN ENCICLOPEDIA INFORMATICA

EDICIONES NUEVA LENTE

RUF ART PICKUP DROP VGUN HOLOGRAM


SPEEDKING
KONIX
© 1986

SUMARIO

Una publicación:

Ediciones Nueva Lente, S. A.

Director editor: MIGUEL J. GOÑI

Director de producción: SANTOS ROBLES.

Director de la obra: FRANCISCO LARA.

Colaboradores: PL/3 - MANUEL MUÑOZ - ANGEL MARTINEZ - MIGUEL DE ROSENDO - DAVID SANTAOLALLA - SANTIAGO RUIZ - LUIS COCA - MIGUEL ANGEL VILA - MIGUEL ANGEL SANCHEZ VICENTE ROBLES.

Diseño: BRAVO/LOFISH.

Maquetación: JUAN JOSE DIAZ SANCHEZ.

Ilustración: JOSE OCHOA.

Fotografía: (Equipo Gálata) ALBINO LOPEZ y EDUARDO AGUDELO.

Ediciones Nueva Lente, S. A.:
Dirección y Administración:
Benito Castro, 12. 28028 Madrid. Tel.: 245 45 98.

Números atrasados y suscripciones:
Ediciones Ingelek, S. A.
Plaza de la Rep. Ecuador, 2 - 1.°. 28016 Madrid.
Tel.: 250 58 20.

Plan general de la obra:
18 tomos monográficos de aparición quincenal.

Distribución en España:
COEDIS, S. A. Valencia, 245. Tel.: 215 70 97.
08007 Barcelona.

Delegación en Madrid:
Serrano, 165. Tel.: 411 11 48.

Distribución en Argentina:
Capital: AYERBE
Interior: DGP

Distribución en Chile: Alfa Ltda.

Distribución en México:
INTERMEX, S. A.
Lucio Blanco, 435
México D.F.

Distribución en Uruguay:
Ledian, S. A.

Edita para Chile:
PYESA
Doctor Barros Borgoño, 123
Santiago de Chile

Importador exclusivo Cono Sur:
CADE, SRL. Pasaje Sud América, 1532.
Tel.: 21 24 64. Buenos Aires - 1.290. Argentina.

Fotomecánica: Ochoa, S. A.
Miguel Yuste, 32. 28037 Madrid.

Impresión: Gráficas Reunidas, S. A.
Avda. de Aragón, 56. 28027 Madrid.

ISBN de la obra: 84-7534-184-5.
ISBN del tomo 3: 84-7534-187-X.

Printed in Spain
Depósito legal: M. 27.605-1986

Precio de venta al público en Canarias, Ceuta y Melilla: 940 ptas.
Septiembre 1986.

Tratamiento de la información

Estructuración de un sistema de proceso de datos



Toda información, desde el punto de vista del ordenador, puede ser clasificada como instrucción o dato. En este último caso se pueden distinguir dos tipos de datos: los numéricos y los no numéricos.

Desde que el responsable de un proceso manual de datos decide proceder a su mecanización hasta que el sistema está dispuesto para entrar en funcionamiento, las instrucciones y normas que rigen la gestión deben ser tratadas adecuadamente para que el producto final sea aceptable.

El prinicipal problema que se encuentra a la hora de diseñar una aplicación es la falta de entendimiento entre el usuario no informático y el técnico.

El primero suele creer que, prácticamente sin ningún esfuerzo, el ordenador resolverá todos sus problemas, incluso los que no tiene previstos en su sistema manual. Y, por su parte, el técnico suele emplear una terminología desconocida para el usuario. El fruto de este desorden suele ser una aplicación que no resuelve el problema que tenía el usuario, sino el que imaginó el informático. El retoque posterior necesario va en detrimento del precio y de la calidad del sistema mecanizado.

En realidad la solución a este problema es sencilla: el usuario debe concienciarse de que el ordenador sólo realizará las tareas que previamente él haya descrito rigurosamente. La ventaja que le aportará será la posibilidad de realizar trabajos que manualmente no podría hacer o que le ocuparían mucho tiempo.

Por otra parte, el informático debe huir de utilizar para la comunicación con el usuario un lenguaje técnico y abandonar la idea de que la programación es un arte de privilegiados. No debe, por tanto, confiar en su inspiración, sino seguir una metodología suficientemente constrastada para llevar a cabo el análisis del problema. No es el objetivo de este capítulo el describir exhaustivamente ninguna de las muchas metodologías existentes para el análisis de aplicaciones, pero sí reflejar los pasos más usuales para la estructuración de un sistema de proceso de datos:

- *Análisis funcional*

En esta primera etapa el objetivo se reduce a definir claramente las tareas a realizar. Hay que proceder a una agrupación funcional de los procesos necesarios para solucionar cada problema, sin entrar en detalle de cómo son dichos procesos.

La colaboración entre el usuario y el técnico debe ser estrecha y concienzuda, ya que el producto de este análisis determinará la calidad final de la aplicación.

- *Análisis orgánico*

A partir de la documentación derivada del análisis funcional hay que realizar el análisis orgánico. En esta fase se estudia con detalle y por separado cada uno de los procesos necesarios, llegando a producir los algoritmos, organigramas y demás descripciones que caracterizarán a cada programa de la aplicación. El resultado final de este análisis orgánico es doblemente útil: por un lado, sirve de nexo entre el usuario y el especialista informático; por otro, servirá de base para el siguiente paso del desarrollo.

- *Programación*

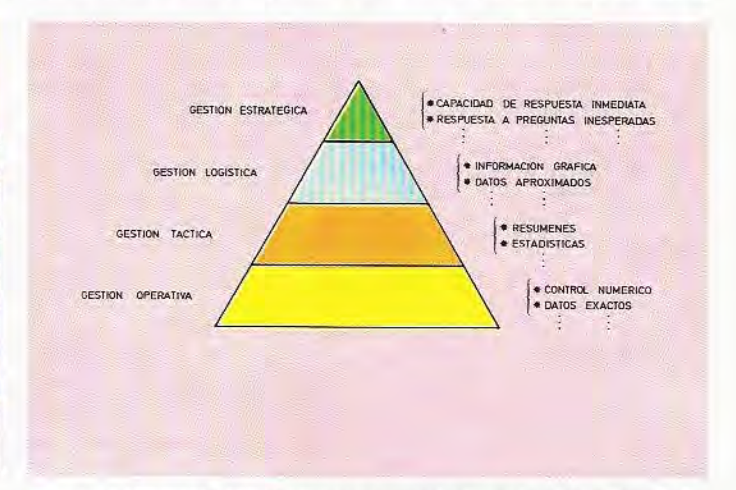
Sólo después de haber realizado el análisis completo de la aplicación se debe empezar a programar. La codificación de los programas debe ser un fiel reflejo de las direcciones del análisis orgánico. Puede afirmarse que la labor del programador se reduce a traducir a un lenguaje de programación las especificaciones que recibe.

- *Prueba*

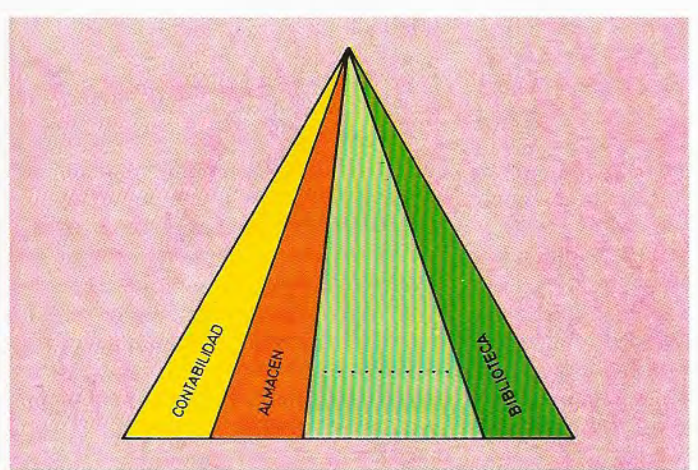
Una vez terminados todos los programas, hay que comprobar que los efectos que producen son los esperados. Los niveles de prueba deben ser dos: primero se verificará el funcionamiento



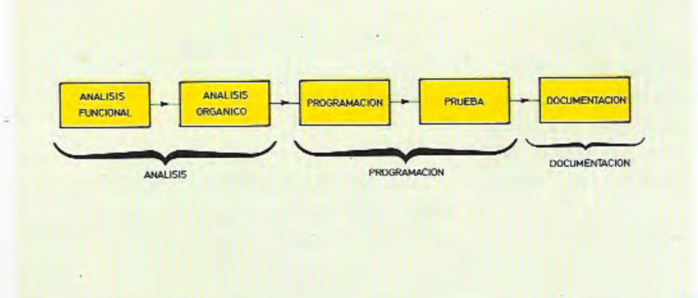
Uno de los problemas más graves que pueden surgir a la hora de mecanizar la gestión administrativa de una empresa es la falta de entendimiento entre el usuario y el técnico informático. Las dificultades quedarán salvadas ccn el empleo por ambas partes de una metodología de trabajo suficientemente contrastada.



Las necesidades de un supuesto usuario que pueden ser solucionadas por un sistema informático aparecen representadas en la presente figura. La base del triángulo quiere significar el nivel de complejidad mínimo en la elaboración de la información; mientras que la cúpula define el grado máximo: ayuda a la toma de decisiones, etc.



El triángulo de la figura anterior se puede también dividir verticalmente, con el objeto de representar las aplicaciones informáticas necesarias para la mecanización integral de una empresa.



Las etapas en el proceso de desarrollo de una aplicación pueden sintetizarse en tres: análisis de las necesidades del usuario, programación o transcripción de algoritmos al lenguaje del ordenador y, por último, documentación y prueba de la aplicación, que incluye la entrega de la información al usuario.

E
ACTUALIZACION FICHERO DE DATOS
LISTADO DE CONTROL DE ERRORES
¿ CORRECTO ?
NO
SI
LISTADO ALFABETICO
LISTADO DE PRESTAMOS
F

Secuencia simplificada de los pasos que en teoría deben darse en la etapa de «análisis funcional» de una aplicación de gestión bibliotecaria o archivo de documentación.

de cada uno de los programas por separado y a continuación el de toda la aplicación en su conjunto.

- *Documentación*

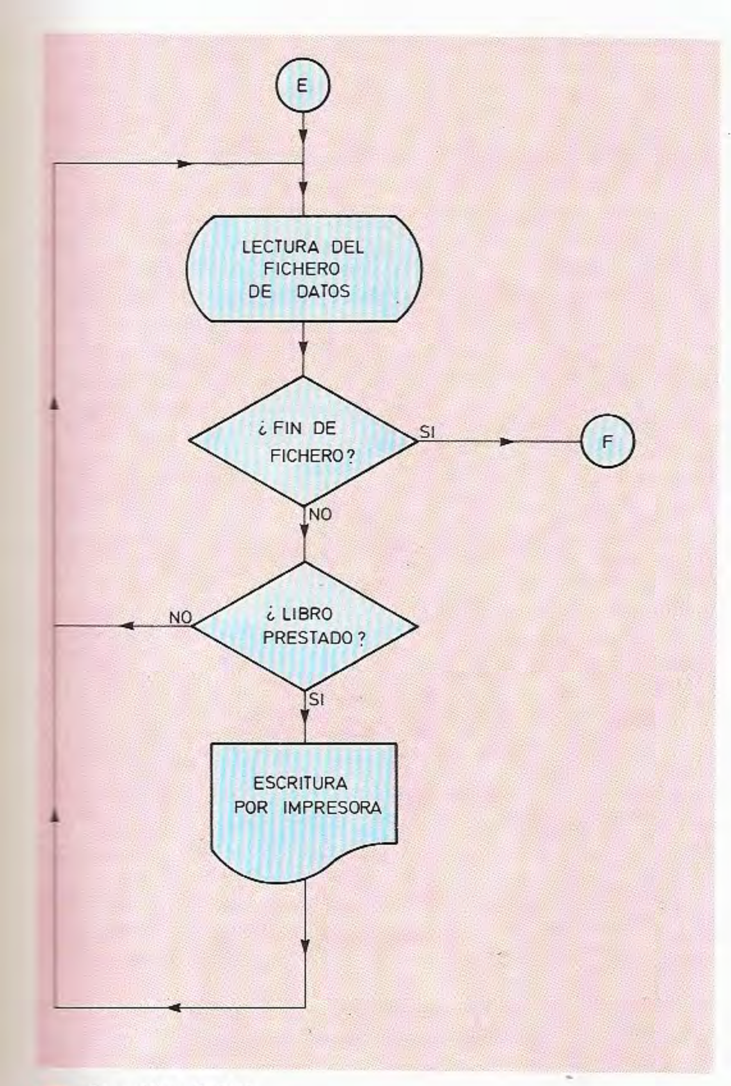
La última fase del desarrollo de la aplicación debe ser su documentación, en la que se incluirán todas las descripciones necesarias para que el usuario sea capaz de utilizar el sistema de forma autónoma, y además para que las futuras modificaciones a realizar en la aplicación puedan ser efectuadas por otros técnicos informáticos. Dado que ambos objetivos son muy dispares, se suelen confeccionar dos manuales distintos: el del usuario y el del programador.

La información numérica

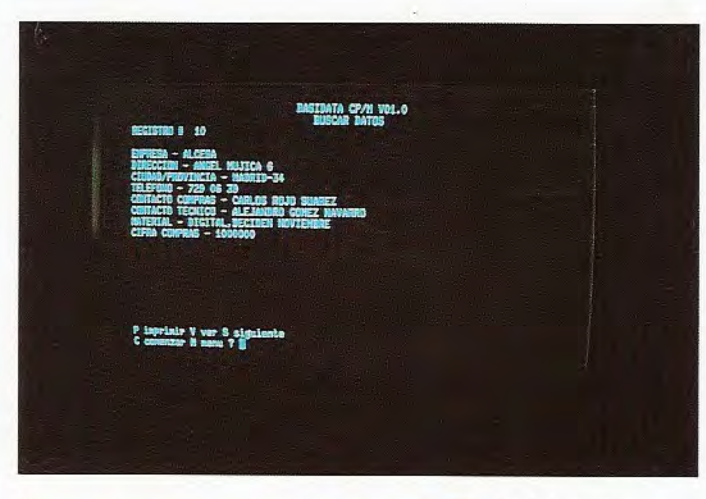
La información numérica sirve, en la mayoría de los casos, para el cálculo de expresiones aritméticas; su tratamiento se puede describir fácilmente: en primer lugar se escribirá el algoritmo que procesará la información. En algunos casos, antes de ejecutar el proceso aritmético propiamente dicho, se puede realizar un control de errores. Por ejemplo: si el dato de entrada se va a utilizar como divisor de otro dato numérico, se debe verificar que es distinto de cero, ya que el resultado de dividir cualquier número entre cero es infinito y esto producirá un error en la ejecución del programa.

También es corriente que la información numérica, tanto si corresponde a datos de entrada como de salida, sea almacenada en alguna memoria auxiliar para su posterior utilización.

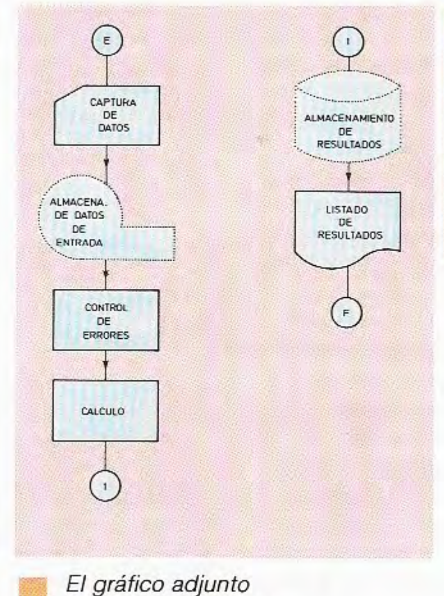
En resumen, podemos generalizar el tratamiento de la información númerica en los siguientes puntos:



El gráfico ilustra el esquema simplificado de la etapa de «análisis orgánico» para la misma aplicación de la figura anterior.



La fotografía muestra un ejemplo de cómo aparece en la pantalla del ordenador un registro cualquiera de una base de datos.



El gráfico adjunto muestra el esquema básico para el tratamiento de la información numérica descrito en el texto.

- Captura de datos de entrada.
- Almacenamiento de datos de entrada (opcional).
- Control de errores.
- Cálculo.
- Almacenamiento de resultados (opcional).
- Devolución de resultados.

La información no númerica

A la hora de resolver un problema mediante la ejecución de un programa en un ordenador es corriente tratar información númerica y no númerica (letras, caracteres especiales, etc.).

Los procesos efectuados sobre este último tipo de información suelen ser de gestión, o, lo que es lo mismo, no se realiza ningún cálculo sobre los datos, sino que se manipulan de alguna otra forma: por ejemplo, se almacenan para su posterior recuperación, etc.

En este caso el tratamiento a efectuar no se puede reducir a la expresión de un algoritmo, y, por tanto, es más compleja la labor de análisis funcional y orgánico.

Dado que el principal proceso de la información no númerica consiste en su almacenamiento y posterior recuperación, se ha trabajado mucho en conseguir sistemas que realicen esa labor eficientemente.

Hace pocos años los bancos de datos eran construidos y manipulados sólo por técnicos informáticos; en la actualidad casi todos los fabricantes de ordenadores disponen de sistemas para generar bases de datos, mucho más eficientes que los anteriores y al alcance de cualquier usuario con una mínima formación previa.

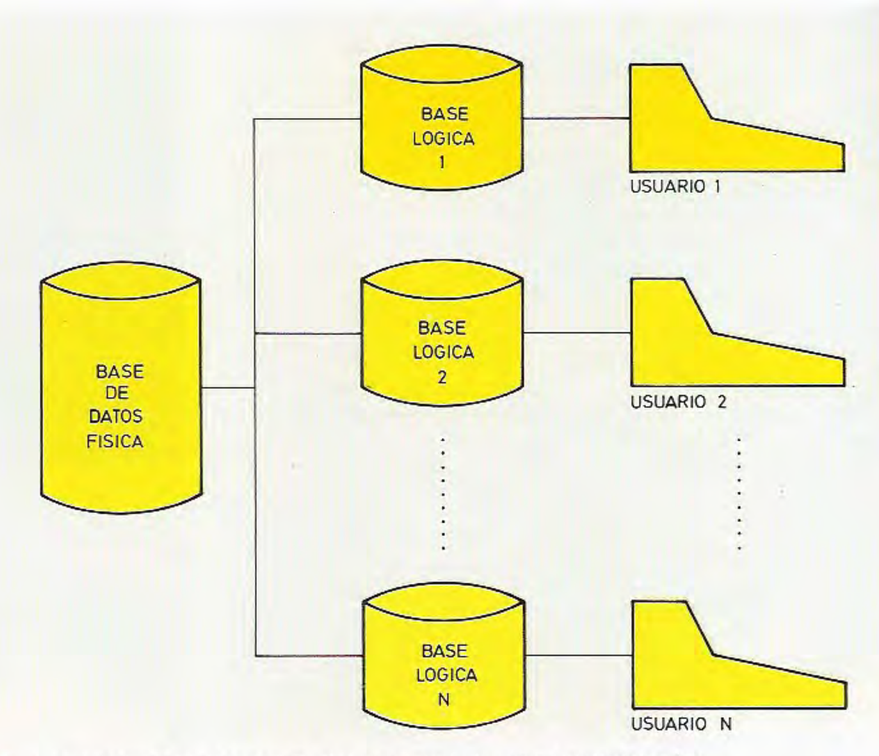
Lógica tri-estado

Hasta ahora se ha hablado exclusivamente de una lógica binaria con dos posibles estados: 0 y 1. En algunos casos, esta lógica es ineficaz para resolver determinados problemas. De ahí surgió la necesidad de utilizar dispositivos que permitiesen tratar un nuevo estado: la desconexión del elemento lógico. En definitiva, un dispositivo tri-estado es aquel que es capaz de trabajar con tres estados distintos: 0, 1 (lógica binaria) e inactivo. La forma de conseguir el estado inactivo es lograr una alta impedancia de salida que provocará la desconexión del dispositivo.

Un ejemplo básico del empleo de la lógica tri-estado en los microordenadores lo constituyen los buses de datos y de direcciones. Ambos son compartidos por las distintas unidades del microordenador y si en algún instante, en cualquiera de los buses existe una superposición de configuraciones procedentes de distintas unidades funcionales, se obstaculizará la correcta interpretación de la información.

La solución a todos estos problemas estriba en controlar el acceso al bus de forma que, en cada instante, tan sólo haya una información presente en el mismo. Para ello el acceso al bus se realizará a través de dispositivos de lógica tri-estado, con lo que, aunque la interconexión física sea permanente, las salidas de las unidades no implicadas en la operación quedarán bloqueadas en estado inactivo (tercer estado).

La estructura de los dispositivos de lógica tri-estado es semejante a la de un operador convencional, pero con una entrada adicional de control. Si esta entrada se encuentra desactivada, el funcionamiento es análogo al propio de la lógica binaria. Pero cuando se activa la entrada de control, la salida adquiere un estado de alta impedancia que bloquea la propagación de información binaria. Por medio del control tri-estado, el microprocesador puede emplear un bus o desconectarse del mismo para ponerlo a disposición de otras unidades. En el caso del bus de direcciones, con un único control tri-estado se resolverá el problema, ya que la transmisión de información se realiza en un solo sentido. En cambio, para el bus de datos es necesario un doble control, ya que su acoplamiento con el microprocesador es bidireccional.



Los sistemas de bases de datos convencionales permiten a los distintos usuarios acceder a la información en la forma que les resulte más útil y práctica.

Para saber más

¿Quién es el responsable del desarrollo de una aplicación informática?

El usuario y el técnico informático. El usuario debe detallar todos los procesos necesarios para resolver su problema, y el jefe del proyecto informático cuidará de la calidad técnica de los programas y de la adecuación de los mismos a los requerimientos del usuario.

¿De qué manera se debe realizar la comunicación entre el usuario y el técnico informático?

Las reuniones de trabajo entre ambos deben ser periódicas durante todo el tiempo que dure el desarrollo de la aplicación y en ellas se debe seguir una metodología concreta.

¿En qué consisten las metodologías para el desarrollo de una aplicación?

Establecen un lenguaje común entre el usuario y el informático y se basan en formularios determinados que deben ser rellenados de acuerdo con las características de la aplicación.

¿Cuántas fases hay en el análisis de una aplicación?

Dos. En la primera se realiza un análisis funcional sin resolver detalles de la aplicación. En la segunda se analiza orgánicamente cada uno de los procesos surgidos del análisis funcional.

¿Cuándo termina el desarrollo de una aplicación?

No basta con que los programas queden comprobados para que el desarrollo se dé por finalizado, es necesario probar la aplicación en conjunto: los datos producidos por un programa deben poder utilizarse en otros. Por último, el técnico debe presentar una documentación sobre la aplicación lo más completa posible.

¿Cuál es la diferencia fundamental entre el tratamiento dado a la información númerica y a la no numérica?

En el primer caso suelen emplearse programas de cálculo que realizan las operaciones definidas por un algoritmo, mientras que para el tratamiento de información no numérica los programas suelen ser de gestión y transferencia.

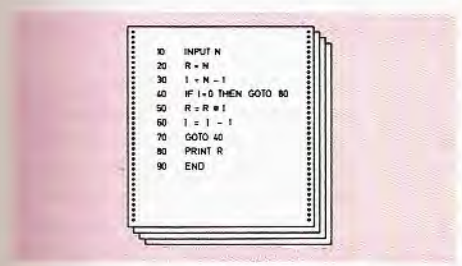
Algoritmos y programas

Vías para la automatización

A lo largo de este capítulo nos ocuparemos de establecer la diferencia entre el razonamiento utilizado para la resolución de un problema (algoritmo) y la implementación de dicho razonamiento en un programa ejecutado por el ordenador. Para no repetir demasiados conceptos ya estudiados en otros apartados de la obra, el tratamiento se limita a una breve descripción de los términos utilizados, para pasar, de inmediato, a resolver dos problemas específicos.

Paso-1	Leer N
Paso-2	R ← N
Paso-3	I ← N − 1
Paso-4	Si I = 0 bifuncar al Paso-8
Paso-5	R ← R × I
Paso-6	I ← I − 1
Paso-7	Bifuncar al Paso-4
Paso-8	Escribir R

Una de las posibles formas de representar un algoritmo consiste en su expresión literal detallada. El algoritmo reflejado en el cuadro adjunto corresponde al cálculo del factorial de un número N.



```
10 INPUT N
20 R = N
30 I = N - 1
40 IF I=0 THEN GOTO 80
50 R = R * I
60 I = I - 1
70 GOTO 40
80 PRINT R
90 END
```

Programa confeccionado en lenguaje BASIC correspondiente a la codificación del algoritmo diseñado para el cálculo del factorial de un número.

Algoritmo

Se llama algoritmo al conjunto de pasos necesarios para la resolución de un problema. Las formas de representar el razonamiento de resolución pueden ser muy variadas: desde la simple descripción literal hasta los organigramas. Todas ellas son válidas siempre que definan las acciones a tomar en todos los casos que se puedan presentar.

Para ilustrar este concepto vamos a dar tres representaciones distintas de un algoritmo que resuelve el simple problema de realizar la suma de dos números.

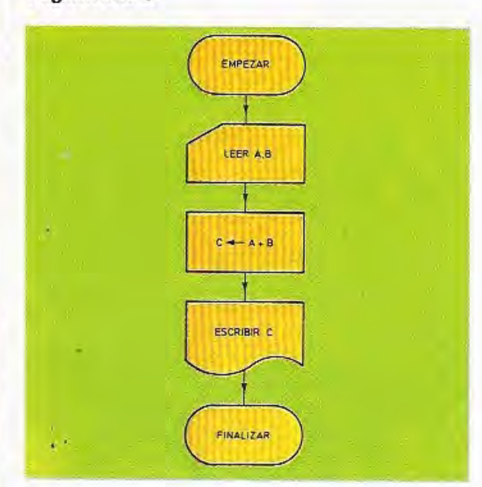
Algoritmo-1

Leer dos números identifcados como A y B. Seguidamente, calcular su suma C y, por último, escribir C.

Algoritmo-2

Paso-1 Leer A
Paso-2 Leer B
Paso-3 C ← A + B
Paso-4 Escribir C

Algoritmo-3



Programa

Un programa no es más que un elemento del conjunto de las posibles formas de representar un algoritmo para la resolución de un problema. La característica común a todos los elementos de este conjunto (programas) es que mediante la representación elegida se puede ejecutar el algoritmo en un ordenador.

A continuación veremos, a título de ejemplo, tres programas en distintos lenguajes que resuelven el problema anterior:

BASIC	PASCAL	FORTRAN
INPUT (A, B) C = A + B PRINT C END	BEGIN READL (A, B) C: = A + B; WRITLN (C); END	REAL (5, 1) A, B C = A + B WRITE (6,2) C 1 FORMAT(2 I 5) 2 FORMAT(I 5) STOP END

El conjunto total de representaciones del algoritmo empleado en el ejemplo debería incluir no sólo los programas redactados en BASIC, PASCAL y FORTRAN, sino en todos los lenguajes y dialectos de ordenador conocidos, tales como APL, COBOL, ADA, ASSEMBLER, etc.

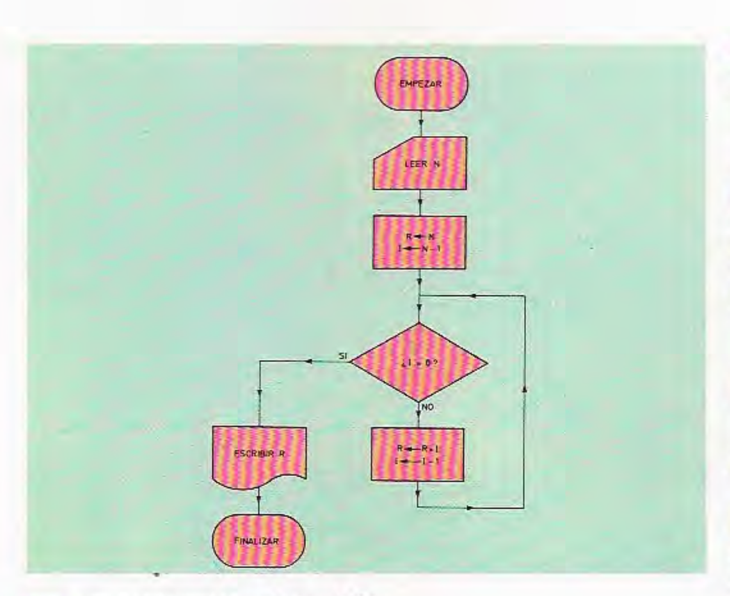
En un caso tan sencillo como éste, parece obvio que lo más eficaz es realizar directamente el programa en el lenguaje de programación elegido. Sin embargo, cuando los problemas son más complicados es muy útil elegir una representación esquemática del algoritmo antes de codificarlo en un lenguaje de programación.

Programa para el cálculo del factorial de un número N (N!)

Este clásico aunque sencillo problema va a permitirnos analizar un algoritmo útil para muchas aplicaciones similares.

Se trata de calcular la expresión siguiente: N! = N × (N — 1) × (N — 2) × ... × 1. Una primera solución intuitiva consiste en fijar el valor de N antes de realizar el programa y posteriormente, con una simple instrucción aritmética, quedaría resuelto el problema. Evidentemente el programa resultante tan sólo servirá para calcular el factorial de un número determinado, con lo que se vulnera una de las primeras leyes de la informática: el programa debe ser independiente de los datos, de forma que, en este caso, sirva para calcular el factorial de cualquier número natural.

La solución consiste en realizar el cálculo mediante un bucle variable, bucle que se repetirá un número de veces acorde con el dato de entrada (N). Si se razona por inducción, tenemos que:



Un algoritmo puede también representarse en forma de diagrama de flujo. El de la figura corresponde al cálculo del factorial de un número N.



El paso previo al desarrollo de un programa específico consiste en la definición de las características del problema o aplicación a resolver para, en consecuencia, redactar los algoritmos a partir de los que se confeccionará el programa.

	Valor inicial	Expresión	Número de operaciones
N = 1	N! = 1		0
N = 2	N! = 2	× 1	1
N = 3	N! = 3	× 2 × 1	2
N = 4	N! = 4	× 3 × 2 × 1	3
...	...		...

En la tabla anterior se observa que el número de operaciones necesarias es N — 1 y que en todos los casos el valor inicial de partida es el propio número N. Ya están, pues, perfectamente definidos los tres primeros pasos del algoritmo:

Paso-1 : Leer N
Paso-2 : R ← N
Paso-3 : I ← N — 1

A continuación habrá que ejecutar un bucle I veces. Para el control del bucle se recurre a un salto condicional: mientras que el valor de la variable I sea distinto de cero se realizarán sucesivos productos; al adquirir esta variable el valor cero significará que ya se han efectuado todas las operaciones, con lo que pasará a escribir el resultado. Así pues, el cuarto paso será:

Paso-4: Si I = 0 bifurcar al Paso ?

Según el número de pasos necesarios para completar el bucle, se sustituirá la interrogación por el valor apropiado. Si se repasa el razonamiento por inducción, podemos observar que el multiplicador de los sucesivos productos a realizar, después de la asignación del valor inicial, coincide con el valor de I; así pues, el siguiente paso será:

Paso-5: R ← R × I

y, a continuación, se resta una unidad a la variable I, ya que el número de operaciones pendientes ha disminuido en una unidad con el paso-5; por tanto, tendremos que:

Paso-6: I ← I — 1

Para finalizar el bucle se incluye una bifurcación condicional al paso-4:

Paso-7: bifurcar al Paso-4

En la secuencia anterior se ha dejado pendiente el número del paso al que se bifurcará cuando I sea igual a cero. Como ya es conocido el número de líneas necesarias para completar el bucle, se sustituirá la interrogación por el número 8, con lo que quedará:

Paso-4: Si I = 0 bifurcar al Paso-8

por último, ya sólo queda por escribir el resultado de las sucesivas multiplicaciones, que tenemos almacenado en la variable R:

Paso-8: Escribir R

En los gráficos adjuntos se incluyen tres representaciones del algoritmo para el cálculo del factorial de un número: la primera es un simple resumen de los pasos enumerados, la segunda consiste en un organigrama que refleja el funcionamiento del algoritmo y la tercera es una representación en lenguaje BASIC y, por tanto, un programa del mencionado algoritmo.

Si se quiere comprobar la eficacia del algoritmo antes de introducirlo en el ordenador, basta con coger lápiz y papel y simular el funcionamiento del ordenador al ejecutar el programa.

En el gráfico correspondiente se comprueba que el algoritmo anterior es correcto, ya que al introducir como dato de entrada 5, produce como resultado 120 que, en efecto, corresponde a 5! = 5 × 4 × 3 × 2 × 1 = 120.

Programa para ordenar una lista de números

Un programa que por su utilidad se incluye normalmente dentro del software de base de un ordenador es el que se ocupa de la clasificación de un conjunto

Algoritmo en expresión literal para la clasificación ascendente de una lista de números.

Paso- 1	Leer N
Paso- 2	Leer T (lista de N elementos)
Paso- 3	I ← 1
Paso- 4	J ← I + 1
Paso- 5	Si T (I) ≤ T (J) bifurcar al Paso-9
Paso- 6	K ← T (I)
Paso- 7	T (I) ← T (J)
Paso- 8	T (J) ← K
Paso- 9	Si J = N bifurcar al Paso-12
Paso-10	J ← J + 1
Paso-11	Bifurcar al Paso-5
Paso-12	Si I = N − 1 bifurcar al Paso-15
Paso-13	I ← I + 1
Paso-14	Bifurcar al Paso-4
Paso-15	Escribir T

Verificación manual del algoritmo para el cálculo del factorial de un número N representado en expresión literal.

PASO	INSTRUCCION	VARIABLES			COMENTARIO
		N	R	I	
1	Leer N	5	—	—	Lectura de «5»
2	R ← N	5	5	—	—
3	I ← N − 1	5	5	4	—
4	Si I = 0 bif. Paso-8	5	5	4	No se bifurca
5	R ← R × I	5	20	4	—
6	I ← I − 1	5	20	3	—
7	Bifurcar Paso-4	5	20	3	Salto incondicional
4	Si I = 0 bif. Paso-8	5	20	3	No se bifurca
5	R ← R × I	5	60	3	—
6	I ← I − 1	5	60	2	—
7	Bifurcar Paso-4	5	60	2	Salto incondicional
4	Si I = 0 bif. Paso-8	5	60	2	No se bifurca
5	R ← R × I	5	120	2	—
6	I ← I − 1	5	120	1	—
7	Bifurcar Paso-4	5	120	1	Salto incondicional
4	Si I = 0 bif. Paso-8	5	120	1	No se bifurca
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
5	R ← R × I	5	120	1	—
6	I → I − 1	5	120	0	—
7	Bifurcar Paso-4	5	120	0	Salto incondicional
4	Si I = 0 bif. Paso-8	5	120	0	Se bifurca
8	Escribir R	5	120	0	Escritura de 120

de números en sentido ascendente o descendente.

Los tres primeros pasos del algoritmo serán:

Paso-1: Leer N (número de elementos de la lista).

Paso-2: Leer T (vector que contendrá los N elementos).

Paso-3: I ← 1.

Para cada bucle basado en el contador I habrá que realizar otro para las comparaciones con elementos situados posteriormente en la lista. Este bucle comenzará siempre en la posición I + 1 y terminará en la posición N; así pues, el cuarto paso será la inicialización de una variable J al valor de I + 1:

Paso-4: J ← I + 1

Después se realizará la comparación entre las posiciones I y J con el operador «≤» para la clasificación ascendente y «≥» para la descendente:

Paso-5: Si T (I) ≤ T (J) bifurcar al paso ?

Si el resultado de la anterior comparación ha sido afirmativo se producirá un



Para saber más

¿Qué diferencia existe entre algoritmo y programa?

Tanto los algoritmos como los programas son descripciones detalladas de los pasos necesarios para resolver un problema. La única diferencia entre ambos conceptos estriba en que en los algoritmos se utilizan símbolos que no son ejecutables por un ordenador y, en cambio, los programas sí son directamente ejecutables por un ordenador.

¿Cómo se representan los algoritmos?

Los algoritmos se pueden representar básicamente de dos formas distintas: mediante una descripción literal, detallando cada uno de los pasos necesarios para resolver el problema, o mediante organigramas.

¿Cómo se representan los programas?

Al igual que los algoritmos, los programas se pueden representar de distintas formas. La característica común de estas representaciones es que deben ser interpretables por el ordenador. Existen muchos y muy variados lenguajes de programación y la elección del más adecuado depende del tipo de problema a resolver.

¿Existe un único programa capaz de resolver un problema?

Para resolver un problema existen muchos algoritmos distintos y, por tanto, también existen muchos programas distintos.

¿Cuándo se puede asegurar que un programa es bueno?

La calidad de un programa se mide mediante dos parámetros distintos: la adecuación de los resultados que produce a los resultados esperados y la calidad de la programación —el programa debe ser sencillo y eficiente.

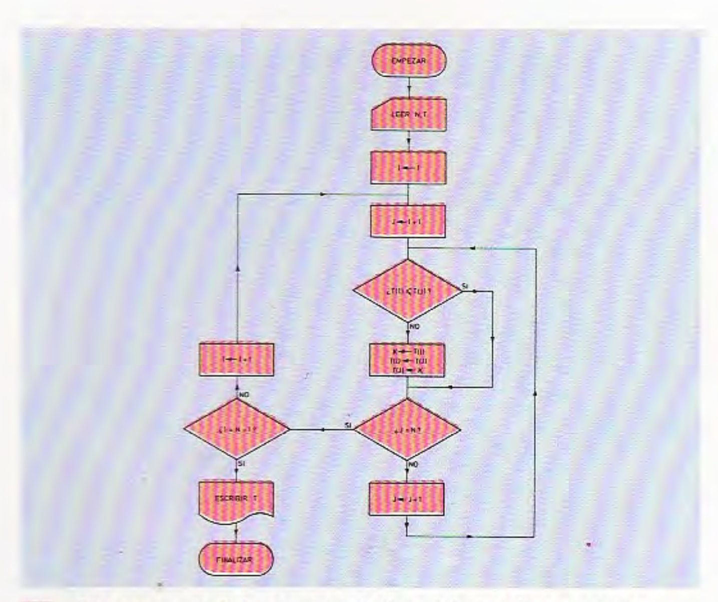


Diagrama de flujo representativo del algoritmo para la clasificación ascendente de una lista de números.

```
10   DIM T (100)
20   INPUT N
30   FOR L = 1 TO N
40   INPUT T (L)
50   NEXT L
60   I ← 1
70   J ← J + 1
80   IF T (I) ≤ T (J) THEN GOTO 120
90   K = T (I)
100  T (I) = T (J)
110  T (J) = K
120  IF J = N THEN GOTO 150
130  J = J + 1
140  GOTO 80
150  IF I = N − 1 THEN GOTO 180
160  I = I + 1
170  GOTO 70
180  FOR L = 1 TO N
190  PRINT T (L)
200  NEXTL
210  END
```

Programa codificado en lenguaje BASIC confeccionado a partir del algoritmo para la clasificación ascendente de una lista de números.

Paso	Instrucción	N	I	J	K	T (1)	T (2)	T (3)	T (4)	T (5)	Comentario
		VARIABLES									
1	LEER N	5	—	—	—	—	—	—	—	—	
2	LEER T	5	—	—	—	6	2	7	4	1	
3	I ← I + 1	5	1	2	—	6	2	7	4	1	
4	J ← I + 1	5	1	2	—	6	2	7	4	1	
5	Si T(I) ≤ T(J) bif P-9	5	1	2	—	6	2	7	4	1	No se bifurca
6	K ← T(I)	5	1	2	6	6	2	7	4	1	
7	T(I) ← T(J)	5	1	2	6	2	2	7	4	1	
8	T(J) ← K	5	1	2	6	2	6	7	4	1	
9	Si J = N bif P-12	5	1	2	6	2	6	7	4	1	No se bifurca
10	J ← J + 1	5	1	3	6	2	6	7	4	1	
11	bif. P-5	5	1	3	6	2	6	7	4	1	
5	Si T(I) ≤ T(J) bif. P-9	5	1	3	6	2	6	7	4	1	Se bifurca
9	Si J = N bif. P-12	5	1	3	6	2	6	7	4	1	No se bifurca
10	J ← J + 1	5	1	4	6	2	6	7	4	1	
11	bifurcar P-15	5	1	4	6	2	6	7	4	1	
5	Si T(I) ≤ T(J) bif. P-9	5	1	4	6	2	6	7	4	1	Se bifurca
9	Si J = N bif. D-12	5	1	5	6	2	6	7	4	1	No se bifurca
10	J ← J + 1	5	1	5	6	2	6	7	4	1	
11	bifurcar P-5	5	1	5	6	2	6	7	4	1	
5	Si T(I) ≤ T(J) bif. P-9	5	1	5	6	2	6	7	4	1	No se bifurca
6	K ← T(I)	5	1	5	2	2	6	7	4	1	
7	T(I) ← T(J)	5	1	5	2	1	6	7	4	1	
8	T(J) ← K	5	1	5	2	1	6	7	4	2	
9	Si J = N bif. P-12	5	1	5	2	1	6	7	4	2	Se bifurca
12	Si I = N − 1 bif. P-5	5	1	5	2	1	6	7	4	2	No se bifurca
13	I ← I + 1	5	2	5	2	1	6	7	4	2	
14	bifurcar P-4	5	2	5	2	1	6	7	4	2	
.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.
4	J ← I + 1	5	4	5	6	1	2	4	7	6	
5	Si T(I) ≤ T(J) bif. P-9	5	4	5	6	1	2	4	7	6	No se bifurca
6	K ← T(I)	5	4	5	7	1	2	4	7	6	
7	T(I) ← T(J)	5	4	5	7	1	2	4	6	6	
8	T(J) ← K	5	4	5	7	1	2	4	6	7	
9	Si J = N bif. P-12	5	4	5	7	1	2	4	6	7	Se bifurca
12	Si I = N − 1 bif. P-15	5	4	5	7	1	2	4	6	7	Se bifurca
15	Escribir T	5	4	5	7	1	2	4	6	7	Escritura resultado

Comprobación manual del algoritmo para la clasificación en orden ascendente de una lista de números.

salto en la secuencia y no se realizará ninguna alteración en la lista. En caso contrario no se bifurcará y se seguirá en secuencia, realizando la permutación entre los elementos T (I) y T (J). Para ello se utiliza una variable intermedia, K, con lo que quedará:

Paso-6: K ← T (I)
Paso-7: T (I) ← T (J)
Paso-8: T (J) ← K

La interrogación del paso-5 puede ser ya sustituida por el número 9, con lo que quedará:

Paso-5: Si T (I) ≤ T (J) bifurcar al paso-9

Para finalizar este bucle interno se comprueba si la variable J ha llegado ya al valor N. En caso negativo se incrementará en una unidad y se comenzará de nuevo el bucle desde el paso-5. En caso afirmativo se realizará la comprobación del bucle exterior mediante la comparación de la variable I con N — 1; si ambas son iguales, se finalizará dicho bucle; si son distintas, se incrementará I en una unidad y se repetirá el bucle desde el paso-4. Tendremos, pues:

Paso-9: Si J = N bifurcar al paso-12.
Paso-10: J ← J + 1.
Paso-11: Bifurcar al paso-5.
Paso-12: Si I = N — 1 bifurcar al paso-15.
Paso-13: I ← I + 1.
Paso-14: Bifucar al paso-4.

Por último, sólo falta por escribir la lista ya clasificada:

Paso-15: Escribir el vector T

Análisis de sistemas

Metodología para el diseño de sistemas de información



El análisis de sistemas tiene tres aspectos fundamentales:

1. Revisar los objetivos y características de un sistema de información mecanizable poniendo de relieve la estrategia a seguir:
2. Justificar una planificación adecuada del sistema de información para evitar errores y problemas.
3. Evidenciar la necesidad de emplear una metodología probada y completa que garantice el éxito del sistema.

El objetivo del presente capítulo será, precisamente, describir una metodología de diseño estructurado de sistemas de información.

Objetivos y estrategia de un sistema de información

La información es un recurso corporativo que resulta tan caro como el resto de los recursos consumidos en cualquier empresa, y por ello debe ser planificado, gestionado y controlado para ser más efectivo dentro de la organización. Por consiguiente, los objetivos principales de un sistema mecanizado de información son los siguientes:

1. Conseguir que soporte las necesidades de información a corto, medio o largo plazo.
2. Proporcionar a todos los niveles de la empresa la información necesaria para controlar las actividades de la misma.

La automatización es un medio irrenunciable para conducir a la moderna empresa hacia los apropiados objetivos de organización, eficacia y productividad.

3. Establecer prioridades de información dentro de la empresa, de forma que se satisfagan las necesidades en el mismo orden de su importancia.
4. Conseguir que el sistema de información sea duradero y se adapte a la evolución de la empresa.

La consecución de estos objetivos resulta difícil de conseguir cuando la empresa para la que se desarrolla el sistema de información es medianamente compleja.

Si tenemos en cuenta que en el análisis del sistema intervienen personas de

muy distintas especialidades, es fundamental marcar una estrategia para alcanzar los objetivos antes mencionados.

Algunos de los puntos principales de esta estrategia son los siguientes:

1. Integrar el sistema de información dentro del plan general de la empresa.
2. Hacer que sea el sistema de información el que dependa de los procesos de la empresa, y no viceversa.
3. Conseguir que la organización sea independiente de los datos, evitando así que una modificación organizativa implique la inutilidad del sistema.
4. Establecer una planificación y control central de los distintos subsistemas de información (si el sistema de información es muy complejo puede ser dividido en varios subsistemas, que se estudian por separado).
5. Determinar una única fuente para cada clase de datos y fijar responsabilidades sobre los mismos.

En definitiva, el sistema de información debe garantizar que la información que tenemos es la que queremos, que la información que queremos es la que necesitamos y que la información que necesitamos está disponible.

Metodología de diseño estructurado de sistemas de información

La metodología que vamos a exponer a continuación está basada en uno de los métodos de análisis más contrastados, como es el método de Warnier. Este método parte del conocimiento completo de las necesidades de información, lo cual no es normal; por ello incluiremos una fase previa, no perteneciente a la metodología de Warnier, con la que se pretende proporcionar una ayuda para determinar las necesidades de una manera progresiva y estructurada, evitando olvidos que podrían obligar a posteriores reestructuraciones del sistema.

Una vez conocidas las necesidades de información, se deben establecer los datos que le serán suministrados al sistema desde el exterior y los que el propio sistema elaborará internamente para producir los informes y procesos deseados en cada momento.



El análisis del sistema de tratamiento y almacenamiento de la información debe ser descendente: partiendo de los niveles superiores se ha de llegar a los datos elementales.



En cambio, la implantación del sistema debe ser ascendente y partiendo de los datos elementales se debe llegar, mediante síntesis, a los grandes objetivos y planes.

También se establecerán los procedimientos de actualización y agrupamiento de datos y, por último, se establecerá la sucesión de los diferentes tratamientos necesarios.

En este primer capítulo dedicado al análisis de sistemas nos limitaremos a definir el referencial, es decir, el universo del sistema de información.

Estructura interna del sistema

Todo sistema informático asegura el almacenamiento, la transmisión y el tratamiento de los datos.

El almacenamiento tiene que ser gestionado a partir de movimientos externos al sistema; la transmisión se hace mediante movimientos internos generados por el propio sistema y, por último, es necesario realizar unos tratamientos que producirán los informes y los propios movimientos internos.

En resumen, los datos del sistema constituyen los resultados de los tratamientos de puesta al día y las entradas de los tratamientos para la obtención de los informes solicitados por los usuarios. La definición de los informes solicitados y de los movimientos de entrada del sistema informático constituyen una tarea esencial.

La metodología que estamos describiendo presenta una forma de llevarlo a cabo para garantizar el éxito final del sistema y preparar una documentación adecuada para la posterior programación estructurada de los tratamientos del sistema.

Informes de salida del sistema

Es necesario que los destinatarios de los informes describan personalmente qué información desean recibir y con qué estructura. Precisamente estas descripciones son la base que servirá para que el sistema sea planificado correctamente.

Cuando hablamos de sistema, no nos estamos refiriendo sólo a un ordenador, sino a todo un sistema lógico de almacenamiento y tratamiento de datos.

Para desarrollar el sistema es necesario, como decíamos antes, realizar un exhaustivo análisis de la organización de cada informe.

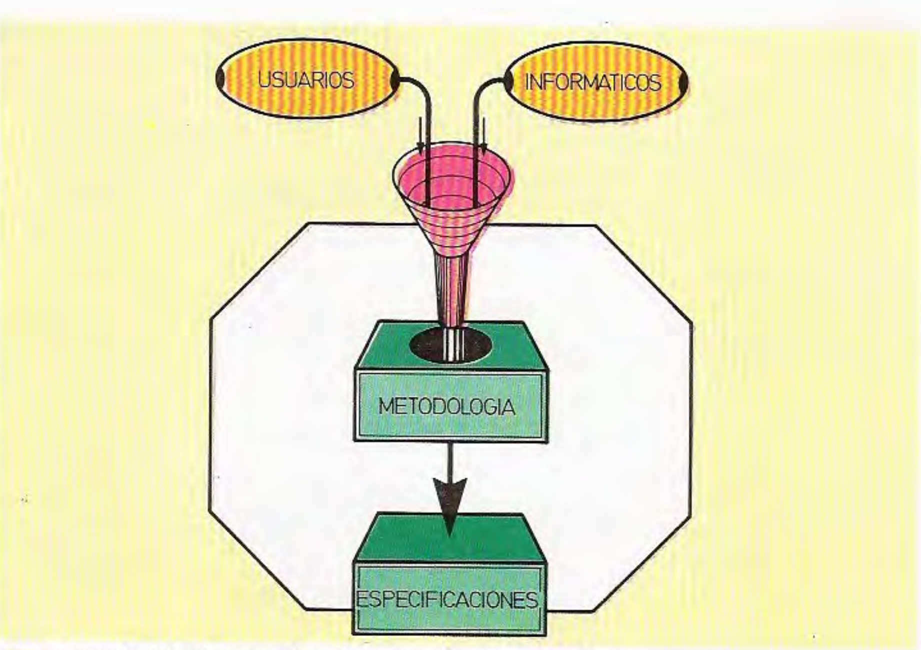
Para facilitar esta organización se pueden utilizar cuatro formularios distintos:

1. *Diagrama jerárquico*

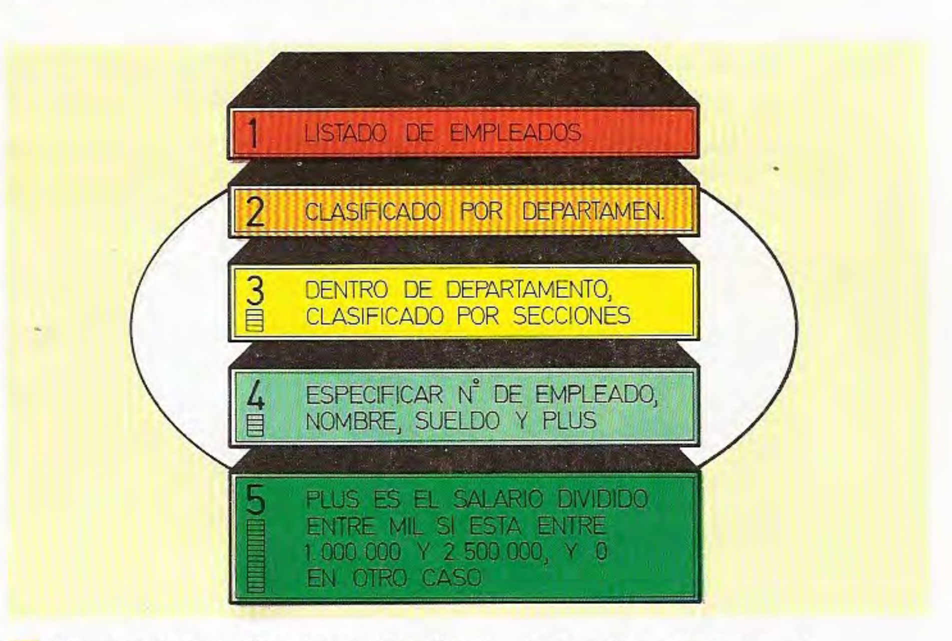
De él se debe desprender fácilmente la estructura del conjunto de datos que se incluyen en el informe. Para ello se debe marcar claramente las dependencias jerárquicas que existan entre ellos.

2. *Modelo de estado*

En este formulario se pretende sumi-



La metodología para el análisis de sistemas produce unas especificaciones que se obtienen tomando en cuenta tanto los datos de los usuarios como del personal informático.



Ejemplo de especificaciones de un listado que se utiliza para ilustrar el contenido de los formularios de solicitud de informes.

Para saber más

¿Cuáles son las principales condiciones necesarias para el éxito de un sistema de información?

1. El método de análisis debe ser descendente; partiendo de los niveles superiores de la empresa, se debe ir bajando hasta llegar a los datos elementales.
2. El sistema se debe implantar ascendentemente mediante análisis.
3. Los datos deben ser manejados como un recurso más de la empresa.
4. El sistema debe estar basado en el análisis de los procesos de la empresa.
5. Contar con la comprensión y el compromiso directo de los usuarios del sistema.

¿Son suficientes las anteriores condiciones para garantizar el éxito de un sistema?

No se puede condensar en una «receta» dichas condiciones, pero es importante destacar que, aunque las condiciones anteriores no son suficientes, sí son necesarias.

¿Quién es más importante en el análisis de un sistema mecanizado de información, el personal informático o el personal usuario?

Ambos colectivos son idénticamente importantes. Para el buen desarrollo del sistema es imprescindible partir de un conocimiento completo de los procesos que se integrarán en el sistema, y esto sólo es posible con la participación del usuario. También es imprescindible que el análisis del sistema tenga en cuenta que al final debe ser implantado en un ordenador, y esto sólo es posible con la participación de técnicos informáticos.

¿Cuántos formularios son necesarios para la descripción de los informes de salida de un sistema informático?

Cuatro: 1) El diagrama jerárquico para establecer la relación entre los datos del informe; 2) el modelo de estado para representar la disposición física de los datos; 3) las especificaciones de datos, con las características generales de los mismos, y 4) fórmulas y condiciones de los cálculos necesarios para obtener datos.

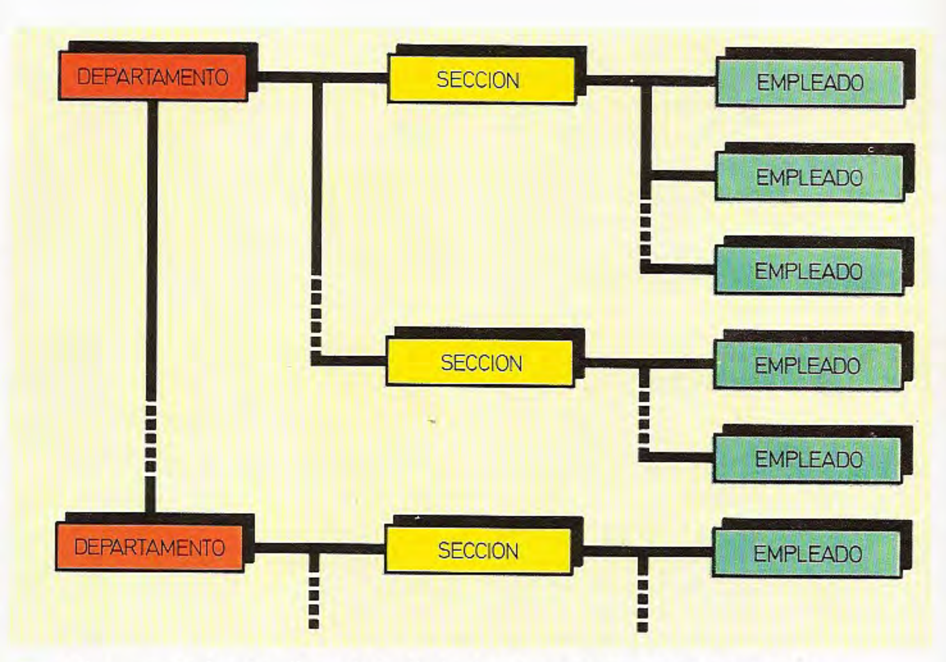
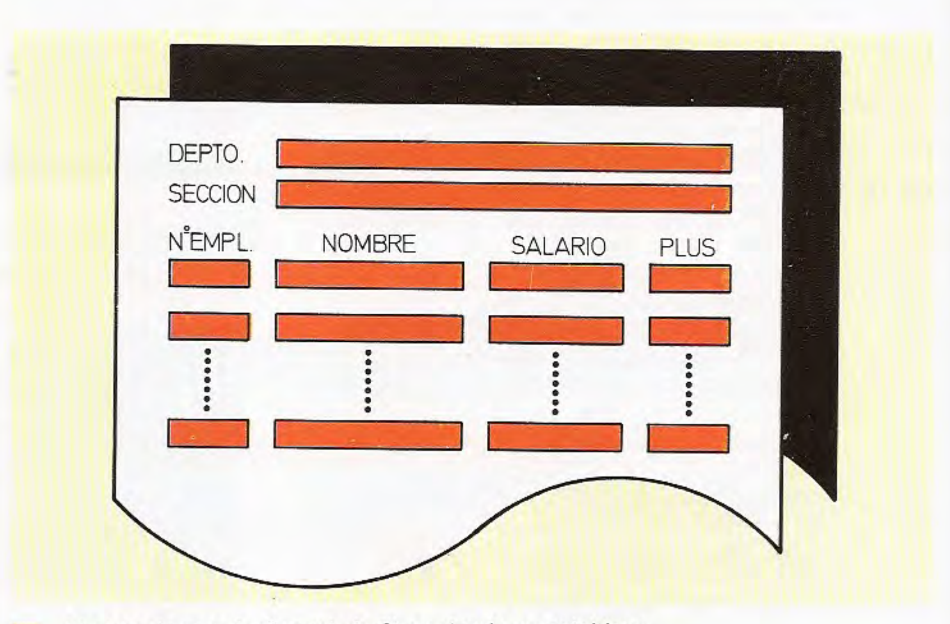


Diagrama jerárquico (formulario 1) de donde se desprende la estructura de los datos que se incluyen en el informe.



Modelo de estado (formulario 2) en donde se establece la disposición física de los datos en los informes.

nistrar la disposición física de los datos en el soporte que los contendrá.

3. *Especificación de datos*

En la primera definición de los datos que aparecen en el informe realizado en el modelo de estado se indica la situación física; además es necesario especificar otras características, para lo cual se utiliza este tercer formulario.

4. *Fórmulas y condiciones*

Cuando alguno de los datos que aparecen en un informe deben ser calcula-

dos, es necesario detallar las fórmulas y condiciones que deben ser tenidas en cuenta en el momento de efectuar los cálculos.

Organización de los datos primarios

El problema planteado es el siguiente: dados unos objetivos y unos informes solicitados por el usuario, hay que diseñar un sistema que cumpla dichos requisitos. Es decir, se debe determinar la organización de los datos necesarios para obtener los informes, la construcción de los programas y la organización de la puesta al día. A todo esto Warnier lo denomina «unidad de realización».

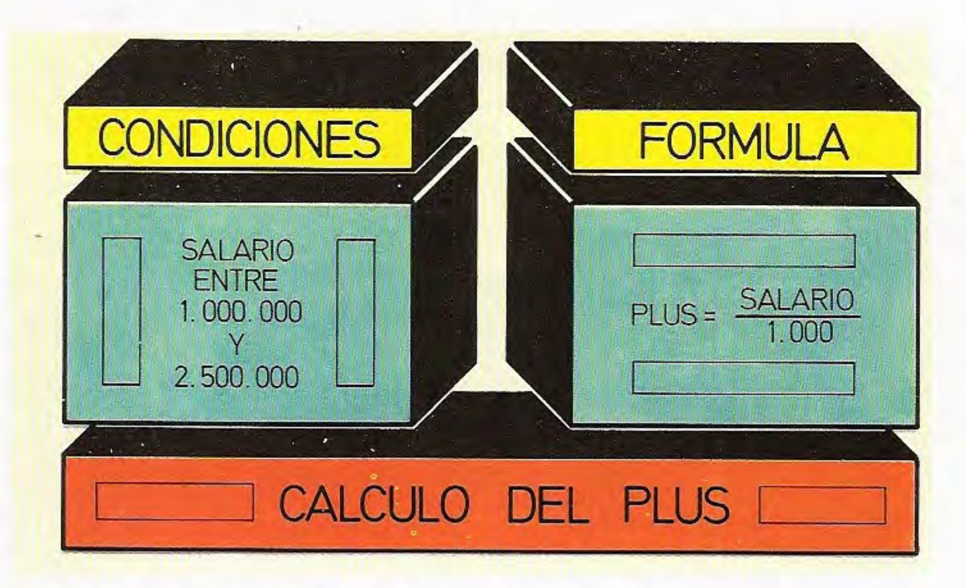
Para realizar la organización de los datos es preciso establecer una correspondencia entre los informes pedidos por los usuarios y el conjunto de todos los datos integrantes del sistema, con el fin de determinar los distintos ficheros o bases de datos en que serán agrupados los datos.

Para establecer la correspondencia citada se puede utilizar un formulario, que a la vez servirá de diccionario del sistema, en el que se reflejará en cada fila un dato y en cada columna un informe, marcando con una «X» las posiciones adecuadas, de forma que al final se puede saber qué datos aparecen en un informe sin más que mirar su columna, o todos los informes que utilizan un dato mirando una única fila.



NUMERO	DESCRIPCION	TIPO	POSICIONES
1	DEPARTAMENTO	ALFABETICO	30
2	SECCION	ALFABETICO	30
3	Nº EMPLEADO	NUMERICO	4
4	NOMBRE	ALFABETICO	40
5	SALARIO	NUMERICO	7
6	PLUS	NUMERICO	7

Especificaciones de los datos (formulario 3). Además de la disposición física es necesario indicar las restantes características de los datos.



Fórmulas y condiciones (formulario 4) que deben emplearse al efectuar los cálculos.

Después de haber construido el diccionario, se tomará la decisión de asignar cada dato a la base de datos (o fichero) que le corresponda según la lógica de operación. No existen algoritmos o procedimientos que garanticen que la reunión de varios datos deba o no ser efectuada; en cualquier caso se tendrán que tener en cuenta características tales como:

— No deben existir datos redundantes.

— La capacidad de almacenamiento de un ordenador es limitada y, por tanto, se debe optimizar la utilización de la memoria.

— El tiempo de proceso es un factor tan importante que a veces obliga a efectuar modificaciones sobre las soluciones aparentemente más lógicas.

— Los criterios de identificación de los datos se deben hacer por las claves de los distintos ficheros; por tanto, habrá que elegir cuidadosamente el identificador óptimo para cada fichero y estudiar la interdependencia entre los distintos ficheros.

DATO	SIGNIFICADO	INFORMES							
		1	2	3	4	5	6	7	8

Para realizar la organización de los datos primarios se puede utilizar un formulario como el de la figura, en el que se especificarán tanto los datos como la relación de éstos con los informes.



En este gráfico se resumen las cuatro etapas fundamentales del análisis de sistemas para el tratamiento de la información.

Los padres de la lógica

La lógica teórica, simbólica o matemática (de las tres formas se las conoce) es una extensión del método formal de la matemática en el campo de la lógica. Los grandes progresos que se han llevado a cabo en las matemáticas desde la antigüedad se han apoyado en parte en el hecho de que se utilizaron lenguajes formales y razonamientos lógicos.

La última aportación de la lógica a la sociedad ha sido su utilización en las ciencias de la computación.

Una de las figuras más interesantes y representativas de la Edad Media fue el español Raimundo Lulio. Se le conocía por el sobrenombre de «Doctor Iluminado». Nació en Mallorca, en 1233, y murió en Bugía (Argelia), en 1315.

Aunque parezca increíble, ya en el siglo XIII se utilizaban elementos lógicos similares a los actuales. Este hombre, que en 1265, tras una crisis espiritual, se convirtió repentinamente al cristianismo, se dedicó durante años a profundizar en los problemas de la mística y de la filosofía, escribiendo libros como *Llibre de contemplació en Dìu, Ars magna y Art de contemplació,* etc.

Después de habitar en diversas ciudades de Europa, como Barcelona, París y Nápoles, fijó su residencia en Roma, aunque posteriormente, siguiendo con su espíritu viajero, continuó viajando por toda Europa apoyado por la Iglesia.

La lógica de Lulio, que expuso especialmente en *Ars Magna,* es un método destinado a demostrar la existencia de Dios mediante unos razonamientos lógicos que formaron una primera e importante piedra en la larga construcción de la lógica actual.

Tuvieron que pasar quinientos años para que Leibnitz (1646-1716) concibiera claramente la idea de una lógica matemática en sentido moderno.

Más adelante, A. de Morgan (1806-1876) y G. Boole (1815-1864) obtuvieron resultados interesantes. Las aportaciones de Boole fueron auténticamente importantes, hasta tal punto que en la actualidad la lógica booleana (álgebra de Boole) sigue estudiándose en todas las escuelas como nociones iniciales y fundamentales de la lógica matemática.

Entre los seguidores de Boole podemos destacar a W. S. Jevons (1835-1882) y, especialmente, a C. S. Pierce (1839-1914), que enriqueció notablemente la joven ciencia.

Los diferentes resultados de sus predecesores fueron recopilados cautelosamente por E. Schroder (1890-1895), que en sus *Lecciones sobre el álgebra de la lógica* ofrece ciertas discrepancias con la línea de desarrollo del propio Boole.

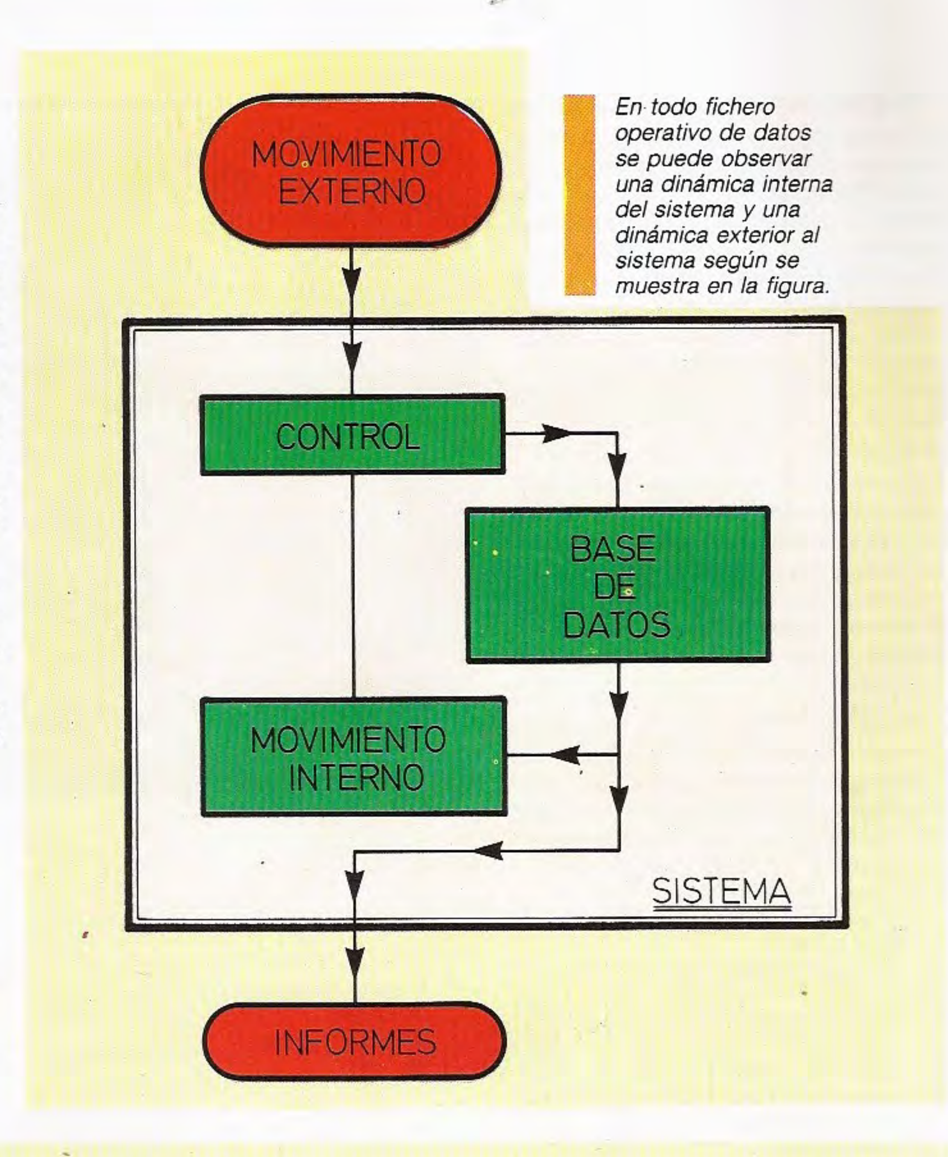
Posteriormente, el matemático G. Frege (1893-1903) publicó trabajos de lógica con una fundamentación más exacta y un tratamiento axiomático más riguroso. Después de Frege surgieron muchos otros lógicos, como G. Peano, A. N. Whitemead, B. Russell, D. Hilbert y P. Bernays. Probablemente, entre todos ellos se puede destacar a Hilbert, que empleó nuevos métodos de cálculo lógico con objeto de llegar, por un camino nuevo, a una reconstrucción completa de la matemática que hiciera ver la no contradicción de los supuestos en que se apoya.

Organización operativa de los datos

En el apartado anterior hemos visto una forma de organizar los datos primarios. El paso siguiente es, conservando la organización de los datos primarios, obtener una solución más «barata», es decir: lo que se pretende es conservar ciertos datos intermedios (secundarios) que permitan obtener los resultados sin necesidad de recurrir a todos los datos primarios. Por supuesto, esta conservación de datos tan sólo se hará cuando suponga un ahorro de espacio y/o tiempo de proceso.

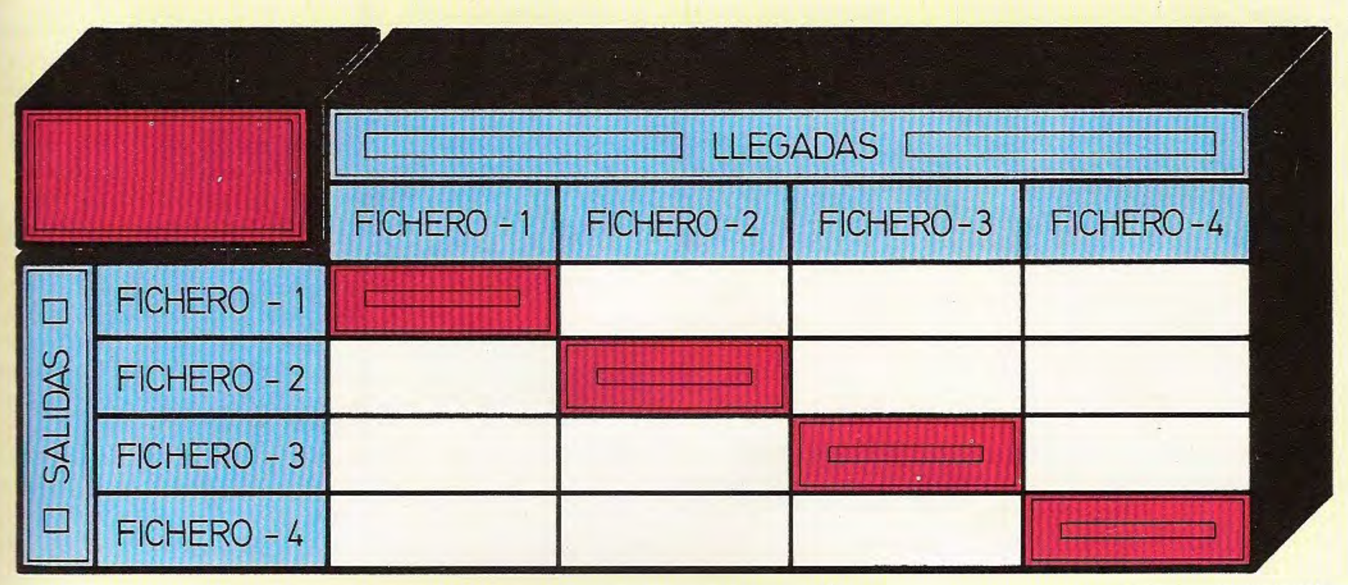
Los procesos para generar datos secundarios son fundamentalmente dos:

1. Procesos de toma de datos y obtención de movimientos internos.
2. Procesos de utilización de movimientos internos y puesta al día de los ficheros.



En todo fichero operativo de datos se puede observar una dinámica interna del sistema y una dinámica exterior al sistema según se muestra en la figura.

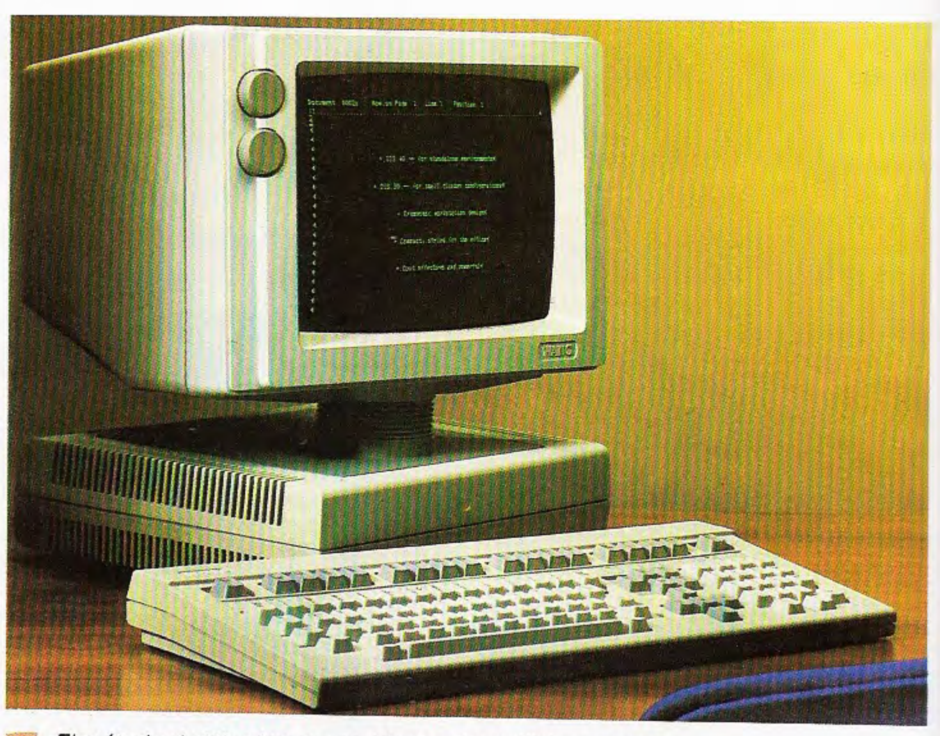
Para el estudio de las interacciones entre los distintos ficheros lógicos del sistema, se suele utilizar una tabla de doble entrada en la que se van especificando las relaciones existentes entre los ficheros.



Mediante esta metodología no se pretende proporcionar una solución definitiva, sino presentar muy claramente las diferentes opciones posibles, entre las cuales se elegirá en virtud de criterios cuantitativos, técnicos o cualesquiera otros, que desbordan el marco de la lógica estricta. En resumen, para elegir una solución entre todas las posibles deben tenerse en cuenta los objetivos de diseño y, al mismo tiempo, se pueden hacer intervenir características técnicas de la máquina disponible.

Una vez dado este último paso, tenemos definidos todos los ficheros lógicos del sistema, tanto primarios como secundarios. Para obtener una solución más operativa habrá que hacer un nuevo estudio partiendo de los ficheros ya definidos hasta llegar a los ficheros lógicos operativos.

Se puede definir como fichero lógico operativo a todo fichero constituido efectuando la reunión de ficheros lógicos, tanto primarios como secundarios, con objeto de tener un menor número de



El método de Warnier es una técnica de análisis que parte del conocimiento completo de las necesidades de informatización, lo cual no acostumbra a ser frecuente.

Para poder efectuar correctamente el análisis del sistema de información de una empresa es conveniente realizar una subdivisión jerárquica de los informes.

DIRECCION GENERAL
- ADMINISTRACION ...
- PERSONAL
  - ORGANIZACION
    - NOMINA
    - CONTROL PRESENCIA
    - RETRIBUCIONES
    - ...
  - RELACIONES
    - AYUDAS SOCIALES
    - FORMACION
    - ...
- PROMOCION ...
- ...



Códigos detectores y autocorrectores de errores

La información, durante la transmisión o almacenamiento, puede sufrir modificaciones involuntarias. Para evitar este problema, y consiguientemente los perjuicios que pudiera ocasionar, existen unos códigos llamados «detectores de errores». La misión de estos códigos es determinar si el mensaje recibido en el extremo de una red de comunicaciones de datos es distinto del emitido.
Además de los códigos detectores de errores existen otros denominados «autocorrectores». Gracias a ellos, los mensajes se corrigen de forma automática, por lo que el receptor puede tener la seguridad de que el mensaje es totalmente fiable.
Para poder realizar la detección o la corrección de errores, el número de dígitos binarios emitidos siempre es superior al estrictamente necesario. Por esta razón, a estos códigos se les llama también redundantes.

Control de paridad

Uno de los códigos detectores más utilizados es el llamado código de paridad de n + 1 bits. Los n primeros bits sirven para codificar cualquiera de las 2^n posibles combinaciones. El último bit, denominado bit de paridad, contendrá un 0 o un 1, según sea necesario para que el número total de unos en el mensaje sea par (si el control de paridad es par). Este código, como ya se indicó anteriormente, sólo sirve para detectar los errores producidos. Por ejemplo, si se elige un control de paridad par, el mensaje 0 1 1 0 1 0 1 0 0 sería correcto, pues contiene un número par de unos, pero el mensaje 0 0 1 0 1 0 1 0 0 sería rechazado por contener un número impar de unos. La desventaja del código basado en el control de paridades es que sólo es parcialmente detector, ya que si en el mismo mensaje se producen un número de errores compensados (0 por 1 y 1 por 0 en igual número de bits, respectivamente), el código no detectará el error.
Los códigos autocorrectores pueden ser muy variados; uno de los más característicos es el denominado código autocorrector de Hamming. Estos códigos pueden llegar a corregir varios errores cometidos en la transmisión de una información digitalizada.

A continuación expondremos el método base para construir un código de Hamming capaz de corregir un solo error.
Supongamos que la información tiene una longitud de «K» dígitos binarios; a éstos se les añadirán «r» dígitos de paridad que serán utilizados para detectar y corregir el posible error. Por tanto, el número total de bits enviados será $n = K + r$.
Si entre el mensaje emitido y el recibido hay un único bit erróneo, tendremos n + 1 posibles configuraciones distintas en recepción: n con un error en cualquiera de los bits enviados y una más sin ningún error.
La forma de determinar «r» es mediante la desigualdad: $r \leq 2^r - K - 1$, a partir de la que se puede construir la siguiente tabla:

K	1	2	3	4	5	6	7	8	9	10	11	12...
r	2	3	3	3	4	4	4	4	4	4	4	5...
n	3	5	6	7	9	10	11	12	13	14	15	17...

Para describir el procedimiento de emisión y recepción de mensajes supondremos que $n = 7$, $K = 4$ y $r = 3$.
El mensaje a emitir será el siguiente:

I_1	I_2	I_3	I_4	C_1	C_2	C_3
información				control		

Una vez decidida la información a enviar, por ejemplo: $I_1 = 1$, $I_2 = 0$, $I_3 = 1$ e $I_4 = 1$, hay que calcular los bits de control, para lo cual se utilizarán las siguientes ecuaciones de control:

$C_1 = I_1 + I_2 \oplus I_3$
$C_2 = I_1 + I_2 \oplus I_4$
$C_3 = I_1 + I_3 \oplus I_4$

Y de esta forma, en el ejemplo anterior tendríamos:

$C_1 = 1 \oplus 0 \oplus 1 = 1 \oplus 1 = 0;$
$C_2 = 1 \oplus 0 \oplus 1 = 1 \oplus 1 = 0,$
y $C_3 = 1 + 1 \oplus 1 = 0 \oplus 1 = 1,$

con lo que el mensaje enviado sería:

1	0	1	1	0	0	1

Supongamos que se produce un error en el cuarto bit de información y el mensaje recibido es:

1	0	1	0	0	0	1

El receptor del mensaje comprobará con las mismas ecuaciones de control los tres últimos bits del mensaje recibido; para ello comparará los resultados calculados por él y los dígitos binarios recibidos; de esta forma se le podrán presentar los siguientes casos:

C_1	C_2	C_3	Diagnóstico
Correcto sin errores	Correcto	Correcto	Mensaje recibido
Correcto	Correcto	Falso	Error en C_3
Correcto	Falso	Correcto	Error en C_2
Correcto	Falso	Falso	Error en I_4
Falso	Correcto	Correcto	Error en C_1
Falso	Correcto	Falso	Error en I_3
Falso	Falso	Correcto	Error en I_2
Falso	Falso	Falso	Error en I_1

En el ejemplo tendríamos:

C_1 recibido $= 0$
C_1 calculado $= 1 \oplus 0 \oplus 1 = 1 \oplus 1 = 0$ $\} \Rightarrow C_1$ corr.

C_2 recibido $= 0$
C_2 calculado $= 1 \oplus 0 \oplus 0 = 1 \oplus 0 = 1$ $\} \Rightarrow C_2$ falso

C_3 recibido $= 1$
C_3 calculado $= 1 \oplus 1 \oplus 0 = 0 \oplus 0 = 0$ $\} \Rightarrow C_3$ falso

Luego se puede deducir que hay un error en I_4 y corregir dicho error sin más que comprobar el diagnóstico asociado a: «CORRECTO - FALSO - FALSO» en la tabla anterior.



Para saber más

¿Cuáles son las principales fases del análisis de sistemas?

1. El diseño de los informes de salida, en donde el usuario indica cuáles son los datos necesarios.
2. La organización de los datos primarios, que consiste en un primer agrupamiento de los datos en ficheros.
3. La organización operativa de los datos, donde se hace una nueva agrupación de los datos atendiendo a criterios de optimización de la explotación.
4. La organización de la puesta al día, que consiste en estudiar los procedimientos que se utilizarán para actualizar los datos del sistema.

¿Qué criterios se siguen para hacer las agrupaciones de datos en la organización de datos primarios?

Evitar las redundancias. Minimizar la cantidad de memoria necesaria para almacenamiento. Disminuir el tiempo necesario para los procesos. Y facilitar la identificación de los datos.

... Y ¿qué criterios se utilizan para realizar las agrupaciones de la organización operativa?

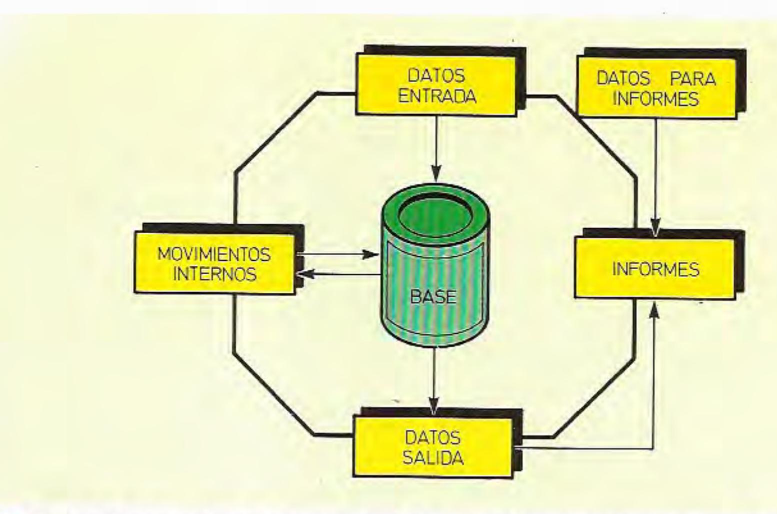
En este caso se hace hincapié en la posibilidad de utilizar datos intermedios de forma que, almacenando estos datos, se facilita la labor de otros procesos que los puedan necesitar.

En la organización de la puesta al día, ¿cuántos tipos de movimientos sobre los datos del sistema se tienen en cuenta?

Dos. Los movimientos internos a los que de alguna manera podemos llamar actualizaciones automáticas, ya que se realizan en función de otros datos ya conocidos por el propio sistema, y los movimientos externos que, como su propio nombre indica, proceden del exterior del sistema.

¿Cuándo es imprescindible utilizar una metodología de análisis como la expuesta aquí?

Depende de la complejidad de la aplicación que se quiera programar. Evidentemente, para realizar un programa de juego no es necesario utilizar esta metodología, basta con realizar un análisis orgánico. En cambio, si en la aplicación intervienen muchos datos distintos, de los que se pretende obtener muchos informes, es imprescindible realizar un análisis completo del sistema antes de comenzar con la programación.



En la actualización de la base de datos se deben considerar cuatro tipos de procesos: los datos de entrada, los datos para informes, los datos de salida y los movimientos internos.

ficheros y una menor redundancia de datos.

Organización de la puesta al día

Todo fichero operativo de datos constituye un conjunto de informaciones que pueden ser:

— Datos de entrada.
— Datos para la obtención de informes.
— Datos para movimientos internos.
— Datos de salida.

Conociendo ya la composición de los registros de cada fichero operativo, se buscará, para cada uno de sus campos, el movimiento interno o externo a partir del cual se efectúa su puesta al día. Evidentemente, las únicas posibles acciones de puesta al día son las siguientes:

— Creación de nuevos datos.
— Supresión de datos antiguos.
— Modificación de datos existentes.

Por definición, todo proceso de puesta al día del sistema está en contradicción con los datos ya existentes y, por tanto, es de suma importancia planificar las actualizaciones adecuadamente para evitar inconsistencias.

Para terminar la descripción de los procesos de puesta al día, vamos a determinar claramente las diferencias existentes entre los movimientos internos y externos.

- *Movimientos internos*

Son datos de entrada para la actualización del sistema obtenidos a partir de otros datos contenidos en los ficheros del propio sistema y que, por consiguiente, actúan sobre los datos secundarios.

En cuanto a las posibles formas en que el sistema puede obtener los movimientos internos, podemos distinguir tres casos:

— A partir únicamente de datos primarios.
— A partir únicamente de datos secundarios.
— A partir de datos primarios y secundarios.

Estos movimientos provocan la actualización en el mismo momento en que se obtienen, o también pueden ser almacenados en ficheros con objeto de proceder a la actualización en el momento oportuno, y una vez realizada la puesta al día no tiene sentido su conservación.

- *Movimientos externos*

Son datos procedentes del exterior del sistema, cuya obtención es responsabilidad del usuario final, mientras que el personal informático se limitará a prever los procedimientos de control antes de efectuar la puesta al día de los ficheros afectados por el movimiento.

Análisis de problemas técnicos y de gestión

Mecanización de procesos



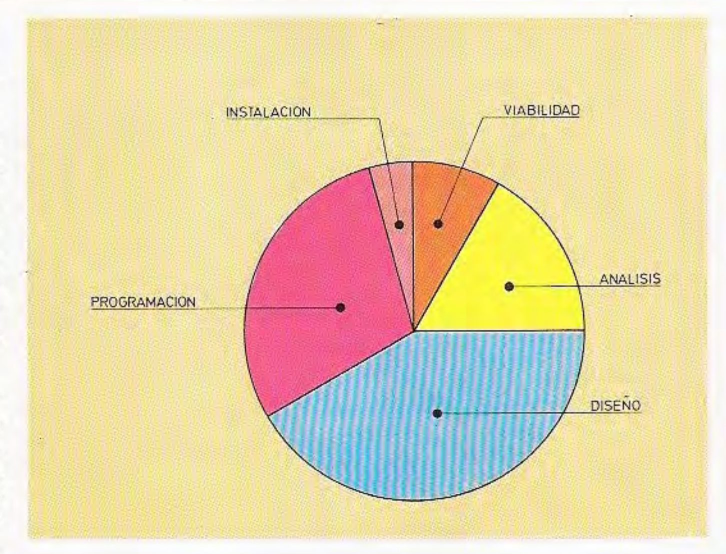
La mecanización mediante ordenador de trabajos complejos, como la contabilidad o la nómina de una empresa, implica la realización de numerosas tareas, que comienzan en el momento en que se toma la decisión de iniciar el proyecto y acaban en el instante en que los programas están funcionando en el ordenador.

Como ya se ha mencionado en un capítulo precedente, estas tareas suelen agruparse en fases o etapas que permiten organizar la secuencia de actividades de todo el personal que interviene en el proyecto.

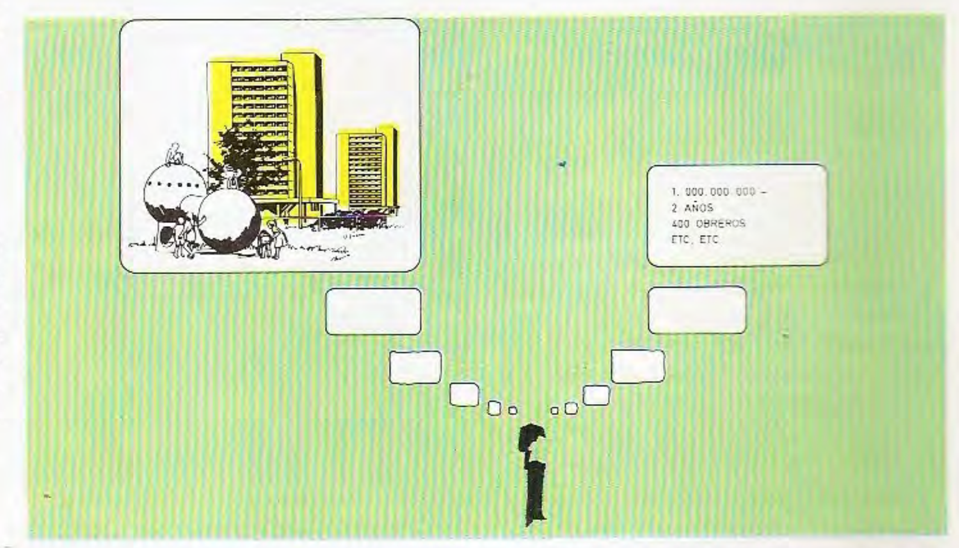
Casi todos los proyectos informáticos comprenden las siguientes fases de realización:

- Fase de estudio de la viabilidad.
- Fase de análisis del problema.
- Fase de diseño.
- Fase de programación.
- Fase de instalación.

A continuación y tras dar un repaso a la naturaleza y objetivo de cada una de estas fases, concretaremos su aplicación al análisis de los dos grandes tipos de problemas que habitualmente se resuelven por medios informáticos: problemas científico-técnicos y de gestión.



La duración de cada una de las fases de elaboración de un proyecto informático son muy desiguales. Como se muestra en la figura, las fases de diseño y de programación son las más largas.





El análisis previo o estudio de la viabilidad es fundamental en la elaboración de un proyecto de cualquier tipo. Un constructor que desee edificar una casa tiene que estudiar primero los costos de realización, el número de trabajadores que serán necesarios, etc.

Fase de estudio de la viabilidad

En esta primera fase se estudia si el problema o trabajo, origen del proyecto, puede ser adaptado o no al ordenador. Luego se analiza el costo y se establece el número de recursos (económicos, de tiempo y personal) necesarios para su realización.

Algunos factores dignos de estudio en esta fase son:

- Ventajas e inconvenientes del proyecto.
- Efectos producidos.
- Personal, tiempo y costos implicados.

Todos estos puntos son tratados por el analista del sistema en compañía del usuario que ha solicitado la ejecución del proyecto. Esta fase es típica de cualquier toma de decisión en nuestra actividad normal. Nadie se embarca en ningún proyecto sin un estudio previo de su factibilidad.

Fase de análisis

Esta fase comienza cuando se toma la decisión de aceptar el proyecto. Está dirigida por el analista, que estudia de forma detallada las informaciones y datos que recibe el usuario y establece cuáles se pueden eliminar y cuáles se deben utilizar en el proyecto.

El analista utiliza diversas técnicas para la recogida de esta información y su posterior análisis. Una de las técnicas más usuales para la recogida de información es la entrevista con todas las personas que vayan a utilizar el proyecto. Estos usuarios pueden aportar sugerencias y requisitos necesarios para un mejor desarrollo. Otra forma de recoger información son los cuestionarios, la investigación personal y la observación del proceso manual.

Esta fase se realiza también en todas las actividades profesionales. Una vez recogidos los datos se procede a un análisis detallado de los mismos, tanto de forma cuantitativa como cualitativa.

Antes de empezar la fase de diseño, el analista y el usuario, o su representante, estudian los resultados obtenidos y se toman decisiones como continuar con el proyecto, cambiar algunos de los objetivos del mismo, cancelar el proyecto, etc.

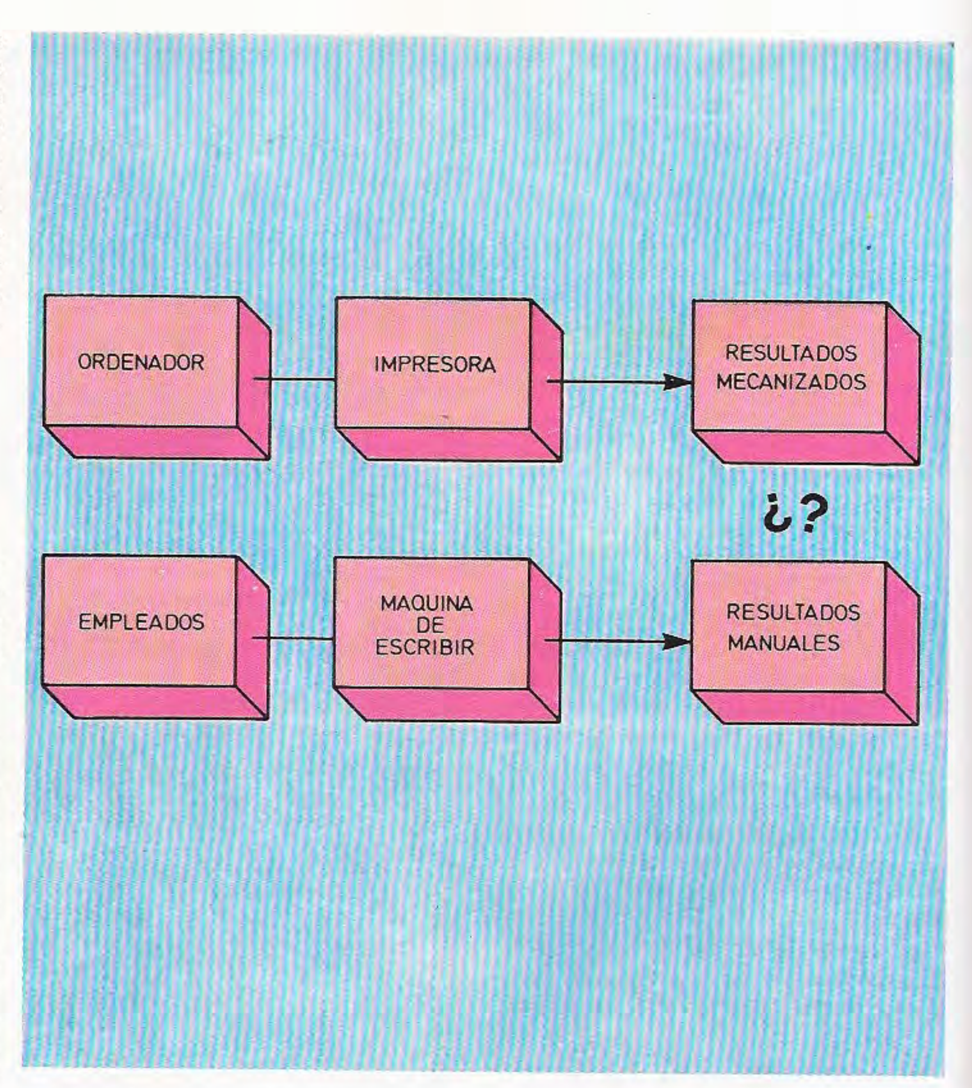
Fase de diseño

En esta fase interviene una nueva persona encargada de continuar el proyecto: es el *diseñador de sistemas,* que se encarga de buscar el tipo de estructuras de los programas o módulos más apropiados para el caso.

La forma ideal de trabajo de un diseñador de sistemas es comenzar el proyecto conociendo los resultados que quiere obtener. A continuación establece los procedimientos necesarios para procesar estos resultados y qué datos de entrada tiene que introducir en ese proceso.

Supongamos, como ejemplo, que una empresa necesita automatizar el pago de la Seguridad Social de sus empleados. El analista de sistemas sabe que tiene que obtener un listado con el importe que cada empleado tiene que pagar a la Seguridad Social. Para llegar hasta ello necesita una serie de datos correspondientes a cada uno de los empleados. Datos típicos en este ejemplo son: si el empleado es soltero o casado, los hijos que tiene, si trabaja su mujer, etc., ya que, de acuerdo con estos datos, la cotización a la Seguridad Social es mayor o menor. A partir de ellos busca las fórmulas y cálculos necesarios para producir la información de salida.

Una vez que el analista conoce estos datos tiene que estructurarlos adecuadamente, de manera que se cumpla el siguiente postulado de la informática: «el sistema perfecto es aquel en el que se toman las medidas adecuadas sobre datos correctos para obtener los resultados necesarios en el momento oportuno».



Los datos necesarios para el análisis de un proyecto informático son de dos tipos: cuantitativos, como los datos de los recibos o el volumen de los mismos, y cuantitativos. Estos últimos datos son, por ejemplo, el organigrama de funcionamiento interno de la empresa que se quiere automatizar.

Para lograr este objetivo el diseñador utiliza también el llamado «análisis jerárquico», que consiste en separar el problema en sus partes componentes. En el ejemplo que hemos visto, el reparto de jerarquías tendría dos niveles. En el nivel superior se encuentra la obtención de listados y de cotización de la Seguridad Social. Y en el segundo, los datos de entrada y proceso. Los datos de entrada se pueden jerarquizar, a su vez, de acuerdo con su importancia. Lo mismo ocurre con el proceso, es decir, se puede dividir el proyecto de forma que aparezcan todas las variables que participan en él.

Una vez que se ha establecido la jerarquía de todos los componentes del sistema hay que «juntar» todas las piezas adecuadamente, creando un diagrama de flujo que enlace a todos los componentes del sistema.

Cuando acaba esta fase de diseño hay que realizar, junto con el analista de las dos primeras fases, una comprobación y ver si el sistema cumple con todas las especificaciones necesarias. Esta comprobación puede llevar a conclusiones

como continuar con la siguiente fase, cambiar algunas especificaciones, abandonar el proyecto, etc.

Fase de programación

En esta fase interviene el programador, que codifica el programa para luego pasarlo a un medio de entrada, como tarjeta perforada, cinta magnética, disco magnético...

Una vez compilados los programas se comienza una etapa de depuración, que consiste en probarlos y corregir los posibles errores.

Los programas se prueban individualmente y se combinan luego en grupos cada vez más complejos. Estos grupos se prueban también conjuntamente. A este fenómeno de agrupamiento de programas se le conoce como «consolidación».

Finalmente se prueba el sistema, emulando el funcionamiento real. Si es correcto, se pasa a la fase de instalación. Si no es correcto hay que intentar solucionar los posibles fallos que se detecten.



Una vez instalado el sistema, comienza la fase de explotación normal del proyecto. Los usuarios del mismo deben recibir una completa documentación: manuales de operación, diagramas del sistema...

Fase de instalación

El nuevo sistema se instala para utilizarlo, durante un tiempo, «en paralelo» con el antiguo, lo que permite la comprobación de los resultados obtenidos.

La «prueba en paralelo» es la única que garantiza que todos los casos reales, tanto generales como particulares, son procesables correctamente por el ordenador.

Con el fin de facilitar a los usuarios completa información de las posibilidades del nuevo sistema y de la forma en que éste va a efectuar su trabajo cotidiano, la empresa o grupo que ha realizado el proyecto recicla al personal que va a manejarlo. Organiza para ello cursos que permitan al personal usuario comprender y resolver cualquier duda que se le presente en el manejo del nuevo sistema.

Una vez instalado el sistema, comienza la fase de explotación normal del proyecto. Los usuarios deberán recibir una documentación muy completa sobre el mismo, que incluye manuales de operación, diagramas del sistema y el correspondiente *dossier* de análisis y programación para facilitar correcciones y revisiones.

El proyecto debe ser revisado periódicamente para incorporar, si es necesario, algunos cambios o comprobar que está cumpliendo perfectamente la función encomendada en el momento de su realización.

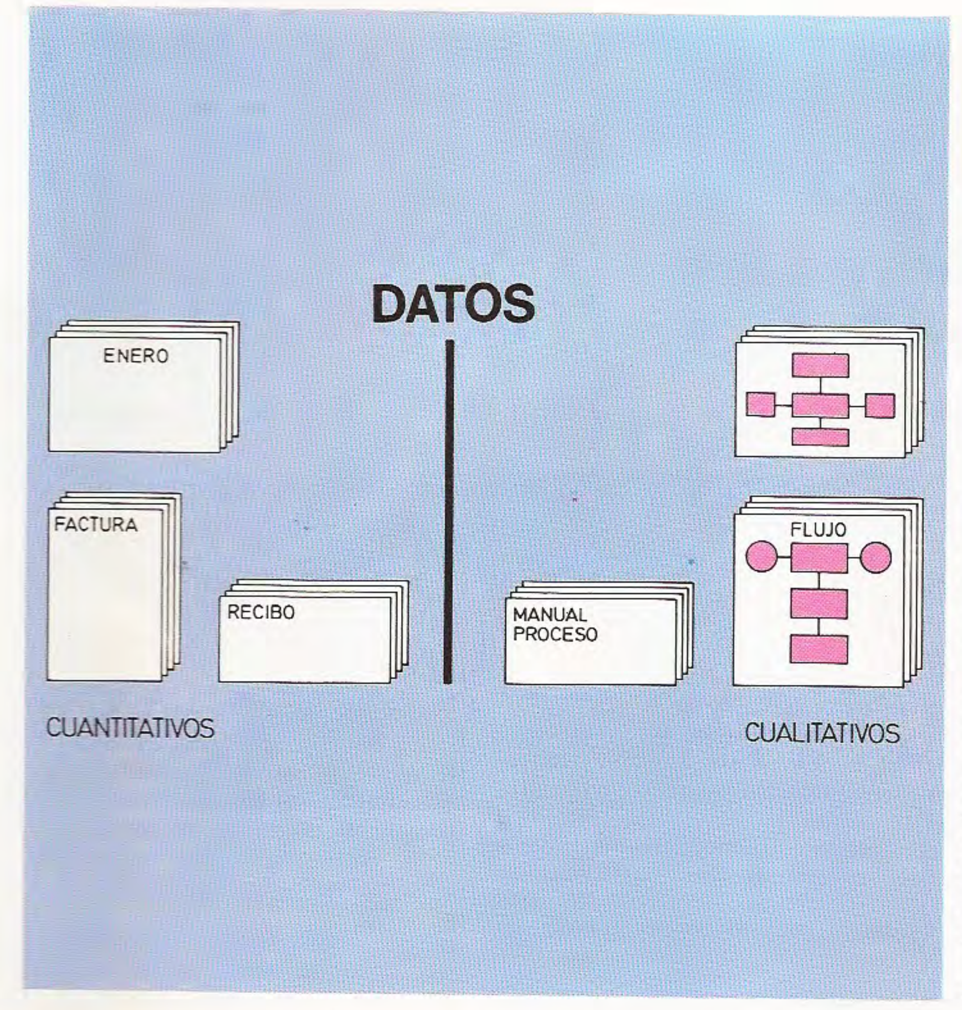
Procesos técnico-científicos

Hay que distinguir entre dos grandes tipos de problemas que se pueden resolver mediante el ordenador: problemas técnico-científicos y problemas de gestión.

Los problemas técnico-científicos son aquellos en los que predominan el número de cálculos a realizar con los datos de partida, sobre el volumen de datos de entrada y salida. Dicho de otra forma, con relativamente muy pocos datos han de realizarse numerosos cálculos. Por otra parte, estos cálculos son mucho más complejos de los que habitualmente se pueden realizar con las cuatro operaciones elementales:

- Resolución de polinomios de cualquier grado.
- Problemas de estadística e investigación operativa.
- Resolución de sistemas de ecuaciones.
- Cálculo de integrales y derivadas.
- Cálculos trigonométricos.

En los problemas técnico-científicos el número de datos de entrada y de resultados es pequeño, por lo que no es ne-



Para probar el buen funcionamiento de un sistema informático se le hace trabajar, simultáneamente y durante algún tiempo con el sistema antiguo, para verificar los resultados obtenidos de ambas formas. A esto se le llama prueba en paralelo.

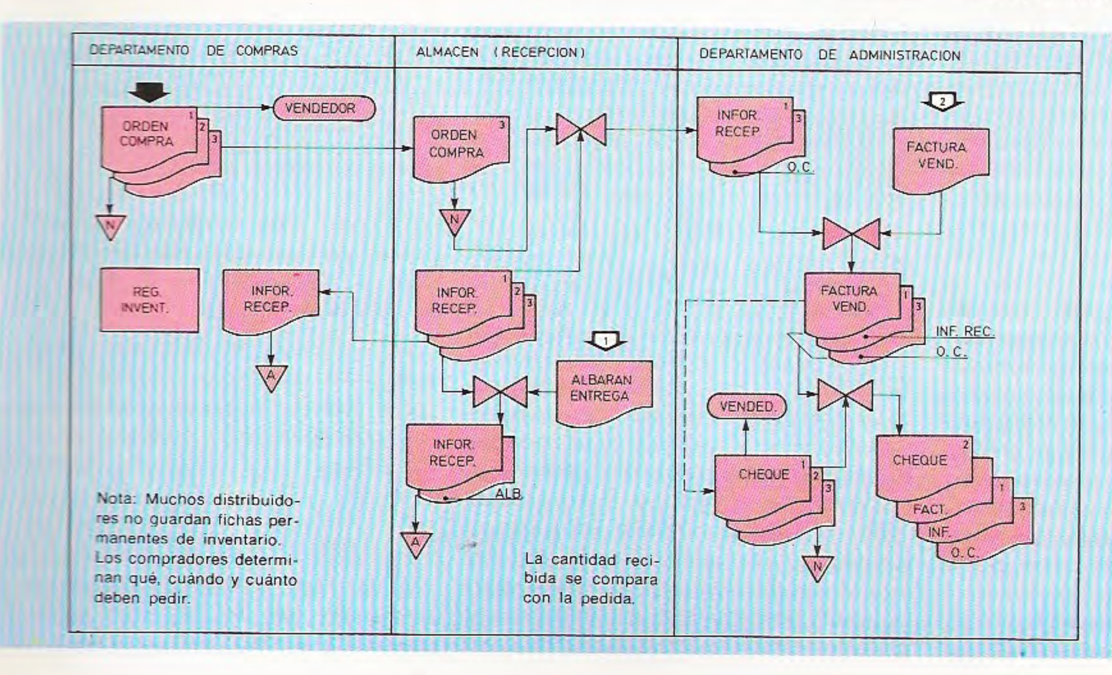
cesario prestar especial atención a la organización de la información de entrada y de salida, aunque a veces se requiere la consulta de tablas bastante complejas.

Las fases del análisis

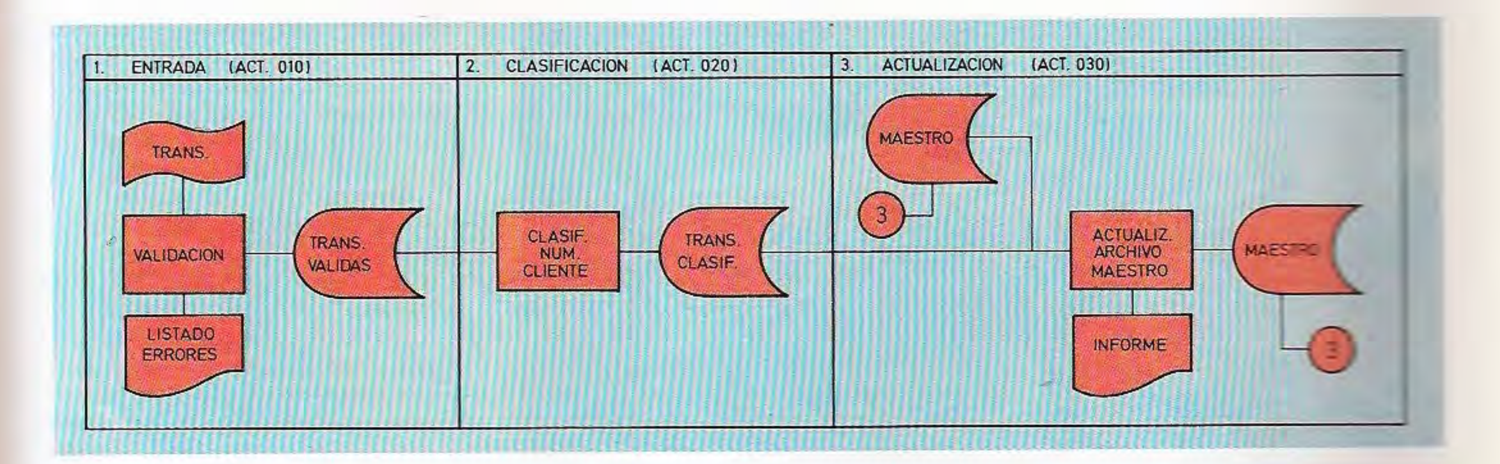
Las tres primeras fases que comprenden el análisis de un proyecto informático son: viabilidad o análisis previo, análisis funcional y diseño o análisis orgánico.

En un problema técnico-científico las fases de análisis son las mismas que en un problema de gestión. Su importancia relativa es generalmente muy diferente, hasta el punto de que casi sólo tiene interés, como tal, el análisis orgánico. Este último se limita, en la mayoría de los casos, a indicar la forma en que se introducen los datos y el formato en el que se desea obtener los resultados.

La fase de análisis funcional, o análisis propiamente dicho, se termina cuan-



Un documento importante para el análisis de un proyecto es el diagrama analítico de los documentos, que muestra el proceso de elaboración seguido por cada uno de ellos.





Los diagramas del sistema muestran la forma en que deben emplearse los archivos y las operaciones a realizar por el programa que se está diseñando.

do se pueden expresar los resultados en función de los datos. Es decir, si representamos por X los valores de entrada y por Y los resultados buscados, cuando tenemos la función

$$\overline{Y} = f(\overline{X})$$

se ha terminado el análisis.

En este momento se debería iniciar la programación en un lenguaje técnico-científico, como el FORTRAN, BASIC, PASCAL, etc. Sin embargo, la función f suele ser muy complicada y puede incluir operaciones que la unidad aritmético-lógica no puede realizar. Esta dificultad obliga a redefinir el concepto de análisis para los problemas técnico-científicos.

Ampliación del concepto de análisis

El ordenador sólo es capaz de realizar las operaciones que resultan de la combinación de las cuatro reglas básicas, por lo que es necesario reducir cualquier cálculo a una combinación de dicho tipo. En otras palabras: integrales, derivadas, límites, etc., necesitan sustituirse por un conjunto de operaciones elementales.

El análisis no puede acabarse, pues, en el momento en que se encuentra la relación funcional f, sino que prosigue hasta que la relación funcional ligue los datos de salida con los de entrada mediante las cuatro operaciones elementales.

El analista necesita, en este caso, de una sólida base matemática dentro del área llamada análisis o cálculo numérico.

Muchas de las expresiones funcionales más utilizadas tienen ya un desarrollo numérico estandarizado, y multitud de fabricantes proporcionan las rutinas correspondientes, por lo que el analista sólo tiene que referenciar la expresión funcional adecuada a fin de que el programador pueda incluirla en el programa.

Introducción al análisis numérico

Todavía no se han encontrado métodos que permitan calcular exactamente muchas de las expresiones funcionales más comunes en los cálculos científicos y de ingeniería: muchos tipos de integrales y ecuaciones diferenciales no tienen una solución analítica conocida. No se renuncia, por ello, a calcular los valores de estas expresiones, sino que se busca una expresión que suministre, para cada caso concreto, un valor todo lo aproximado que queramos y permita determinar una cota del error cometido al tomar el valor aproximado por el exacto.

Estos métodos introducen desarrollos formados por operaciones elementales aritméticas (suma, resta, multiplicación y división) y lógicas (que determinan si el error es admisible).



Para saber más

¿Existen otros métodos de organización de un proyecto informático?

Sí, existen otros métodos de organización, entre los que destacan el método organizacional y el funcional. Pero, básicamente, lo importante es que haya una metodología que facilite el trabajo de todos los que intervienen en un proyecto.

¿Qué significado tiene la palabra «DEBUGGING»?

Es un vocablo inglés que se aplica a la depuración de los programas durante la fase de programación.

¿Qué diferencias existen entre el analista del sistema y el diseñador del sistema?

El diseñador del sistema recoge la documentación elaborada por el analista del sistema durante las fases de viabilidad y de análisis para incorporarla al sistema informático que está proyectado. Es el auténtico «especialista» que hace posible que el proyecto sea resuelto por el ordenador.

¿Qué es lo que se busca, fundamentalmente, al realizar un proyecto informático?

La sustitución de una forma de trabajar manual por un nuevo método mecanizado.
Cuando existe una mecanización previa se busca mejorar el rendimiento del sistema antiguo o la sustitución de un hardware y/o un software anticuado por otro más moderno.



Hay que distinguir entre dos grandes tipos de problemas que se pueden resolver mediante un sistema ordenador: científico/técnicos y de gestión.

El problema que tiene un analista al tratar de reducir su función compleja a un conjunto de operaciones elementales es un problema de análisis numérico. Por consiguiente, los procedimientos que se utilizan en el análisis numérico son los que se usan cuando se resuelven problemas técnico-científicos mediante ordenador.

A la vez, el incremento del uso de los ordenadores ha obligado a un fuerte desarrollo del cálculo numérico, ya que la velocidad de proceso resta importancia al gran número de operaciones que es necesario efectuar por estos métodos y que por medios tradicionales son prácticamente imposibles de realizar. Gracias a este desarrollo hoy día es posible solucionar problemas que hace medio siglo eran inabordables.

Algoritmos

Como ya hemos visto en un capítulo anterior, se llama *algoritmo* al conjunto de operaciones elementales (tanto aritméticas como lógicas) que resuelven un problema determinado. Cuando el algoritmo se expresa en un lenguaje de ordenador tenemos lo que se denomina un *programa.*

Hay que advertir que un mismo pro-

Documentación y personal de un proyecto informático

En la fase de estudio de la viabilidad intervienen el analista y el usuario del proyecto. El usuario presenta el problema y el analista investiga dicho problema de acuerdo con el usuario. Fruto de estas entrevistas, se creará una documentación donde se recojan las ventajas o desventajas del proyecto, sus efectos, personal, tiempo y costos implicados, etc.

En la fase de análisis participa, de nuevo, el analista para investigar a fondo el problema y establecer lo que realmente quiere realizar. El analista se documenta utilizando formularios e informes de los sistemas que se van a utilizar, los rendimientos de los mismos, etc. Toda la información acarreada en esta fase debe ser organizada adecuadamente. Existen diversos sistemas que simplifican la documentación y que comprenden básicamente cinco tipos de formularios correspondientes a la definición de la salida del ordenador, definición de entrada de datos, definición de los cálculos del proceso, definición de procesos lógicos y definición de archivos. El analista puede utilizar los diagramas analíticos.

Cuando el analista y el usuario llegan a un acuerdo sobre las especificaciones del nuevo sistema, éstas se recogen en el formulario o informe de las especificaciones de requisitos. En esta fase el analista actúa como intermediario entre el usuario y el diseñador. En la fase de diseño interviene el diseñador de sistemas (que a veces es el propio analista), quien estructura de forma adecuada los programas. Su labor comienza con el informe de especificaciones y requisitos. Se ayuda de las organizaciones jerárquicas y de los diagramas de flujo para conseguir sus objetivos, y al final proporciona al programador el llamado cuaderno de carga de los programas con todos los datos que se necesitan.

En la fase de programación interviene el programador asesorado por el diseñador del sistema. Codifica y prueba los programas con ayuda del personal que introduce los datos. Las pruebas de consolidación las realiza el programador junto con el analista y el diseñador del sistema. En esta fase el diseñador y el analista preparan la documentación para los usuarios, donde aparecen los medios de archivo, la documentación de las pruebas realizadas, etc.

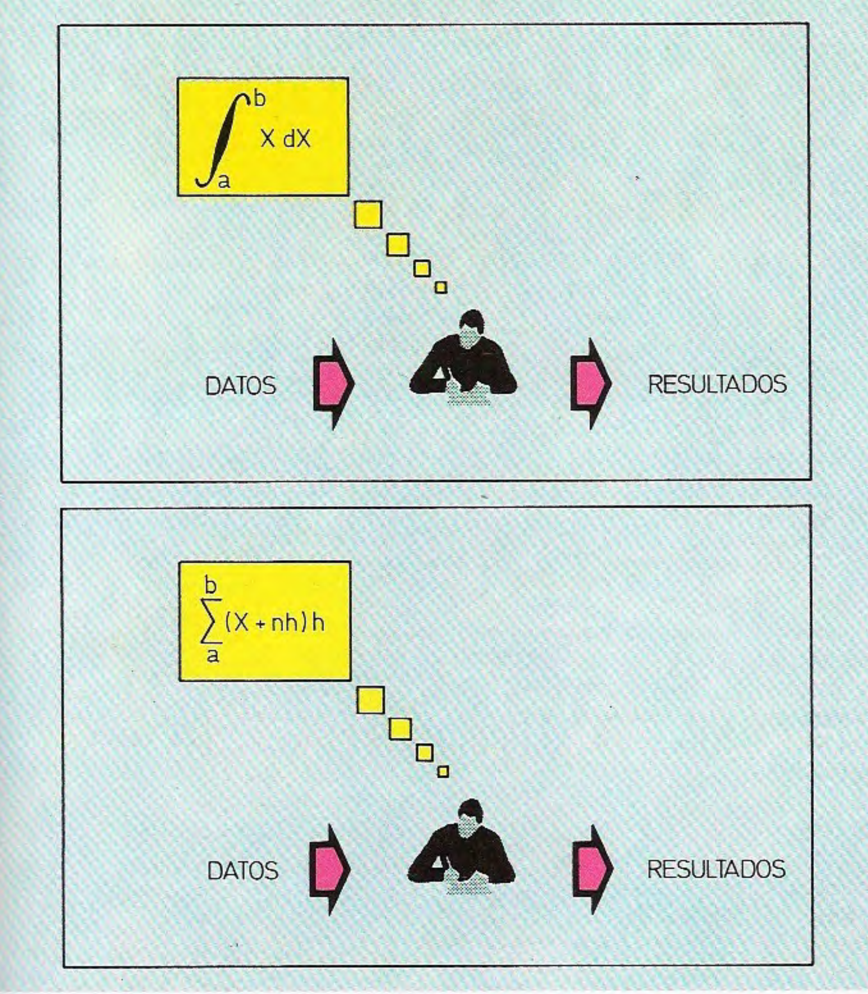
blema puede resolverse mediante algoritmos diferentes. Cada uno de ellos proporciona una solución aproximada al problema. Entre estos diferentes métodos hay alguno que se acerca a la solución correcta con mayor rapidez. Por ello es muy importante determinar, teniendo en cuenta las características del ordenador que se utiliza, cuál es el algoritmo más adecuado para la resolución del problema planteado.

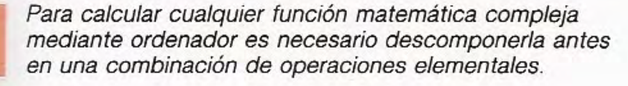
El error de un método es la diferencia entre el valor exacto y el calculado en la ejecución del algoritmo. En general, todos los algoritmos son de carácter iterativo, es decir: repiten una misma secuencia de operaciones. Son cadenas de operaciones aritméticas y lógicas que se ejecutan utilizando el valor obtenido en la operación anterior. Cada vez que se ejecuta el algoritmo la solución obtenida va acercándose al valor real, disminuyendo así el error.

La forma más fácil de representar los algoritmos es mediante un organigrama cíclico o iterativo.

La precisión deseada o, lo que es lo mismo, el máximo error que se acepta como admisible influye mucho en el número de iteraciones a realizar y, por tanto, en el tiempo de proceso. Cuanta mayor precisión se desee, mayor será el tiempo de ejecución.

Hay que señalar que el incremento de tiempo de ejecución no es aritméticamente proporcional a la precisión. Gene-




Para calcular cualquier función matemática compleja mediante ordenador es necesario descomponerla antes en una combinación de operaciones elementales.



Para saber más

¿Existen otras denominaciones para referirse al análisis?

Existen diversas denominaciones, dependiendo de los autores.
Las equivalencias más comunes son:

- Análisis previo: preanálisis, estudio de viabilidad o estudio de factibilidad.
- Análisis funcional: fase de requisitos, requerimientos del sistema o análisis propiamente dicho.
- Análisis orgánico: diseño o creación del sistema.

¿Los problemas técnico-científicos requieren el uso de un ordenador específico?

Las características de los modernos procesadores se adaptan por igual a todo tipo de problemas. Las diferencias entre un sistema de gestión y un sistema para uso científico-técnico se centran, en general, en la periferia del ordenador.

¿Los problemas técnico-científicos requieren algún lenguaje especial?

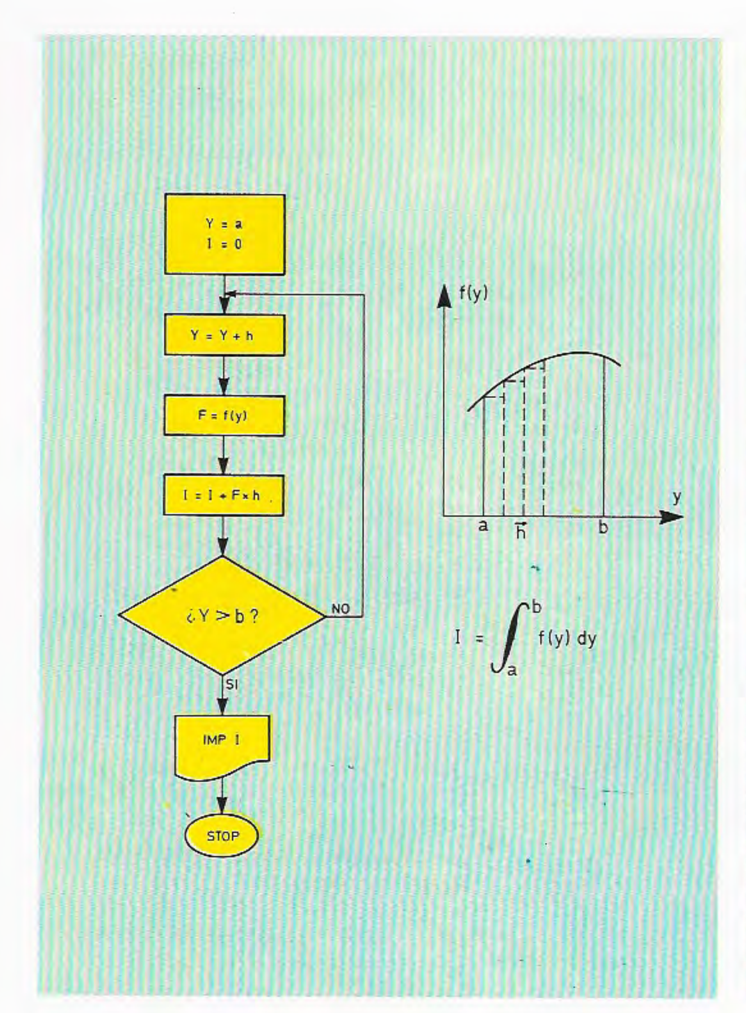
En principio, no. Pero existen lenguajes de alto nivel próximos a los problemas de tipo matemático, como el BASIC, PASCAL, FORTRAN, ALGOL, APL, etc., que facilitan la resolución de este tipo de problemas.

¿Cómo se calculan en un ordenador digital las funciones trigonométricas y trascendentes?

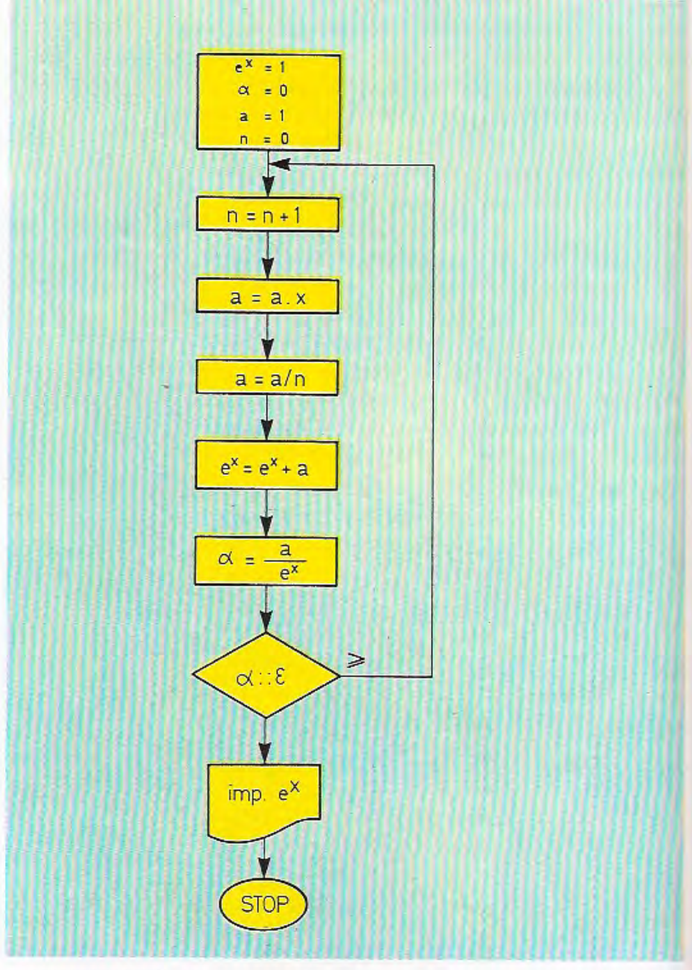
Mediante los desarrollos en serie de dichas funciones. La función se sustituye por una serie expresada en forma recurrente. Cuantos más términos se tomen de esta serie, más se aproxima al verdadero valor de la función, con un error cada vez más pequeño.

¿Por qué es necesario determinar cuándo se termina el cálculo de un desarrollo numérico?

Los desarrollos en serie tienen infinitos términos. Por ello es necesario incluir en el programa una instrucción lógica que permita dar por terminado el cálculo, una vez que el error cometido es suficientemente pequeño.



El valor de una integral se define como el área encerrada por la curva. Uno de los métodos de resolución de integrales consiste en aproximar este área por la suma de una serie de rectángulos de base tan pequeña como se quiera, y de altura el valor de la función en el comienzo de cada pausa.



Organigrama para calcular el valor de e^x con un error inferior a ε, cuyo valor puede hacerse todo lo pequeño que se desee.

ralmente el tiempo es una función exponencial de la precisión: para obtener el doble de precisión se necesita más tiempo del doble necesario para obtener una precisión simple.

En el primer miembro de las expresiones de los algoritmos aparecen los valores a calcular en una iteración; en el segundo miembro aparecen los valores calculados en la fase anterior. El programa deberá disponer de las operaciones lógicas necesarias para que vuelva a repetirse el ciclo iterativo, hasta que el error sea menor que una cota dada o hasta que el número de iteraciones realizadas sea superior a una cantidad prefijada. Esta última condición se impone por el hecho de que algunos algoritmos se acercan a la solución muy lentamente, con lo que el número de iteraciones necesarias para reducir el error es muy grande y, por lo tanto, el tiempo de proceso resulta inadmisible.

Fin del algoritmo

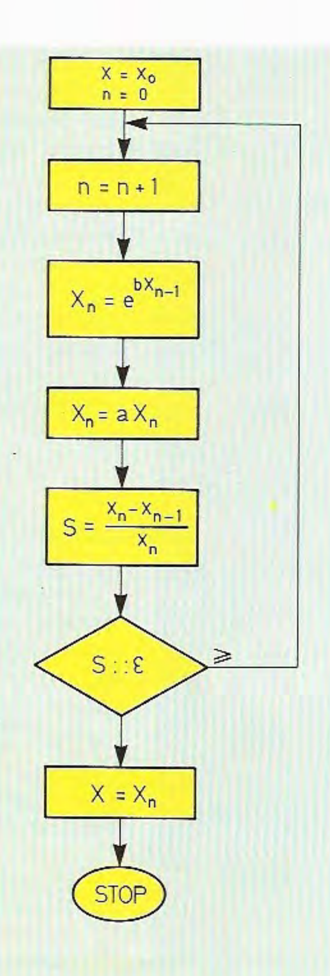
Los criterios más importantes para determinar el final de los cálculos son:

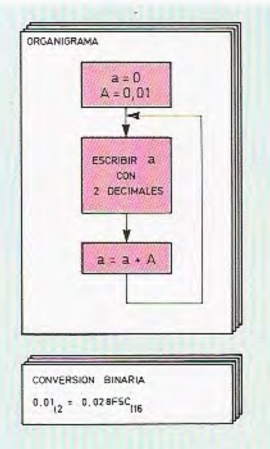
- El cálculo termina cuando la diferencia entre los dos últimos valores calculados es menor a una cantidad preestablecida.
- El cálculo termina cuando el error relativo, es decir, la diferencia entre los dos últimos valores calculados, dividida por el último valor calculado, es inferior a un valor predeterminado.
- Se da por terminado el cálculo cuando el número de iteraciones supera una cantidad definida *a priori*.

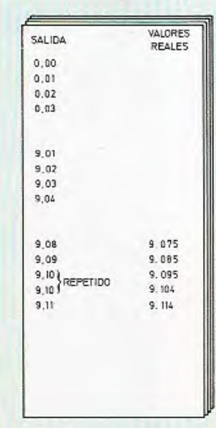
Hay criterios particulares para operaciones específicas del análisis numérico, como en el cálculo de integrales o en la determinación de ceros de polinomios.

Errores de proceso

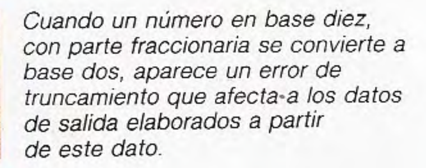
Los problemas técnico-científicos deben ser tratados con particular cuidado para



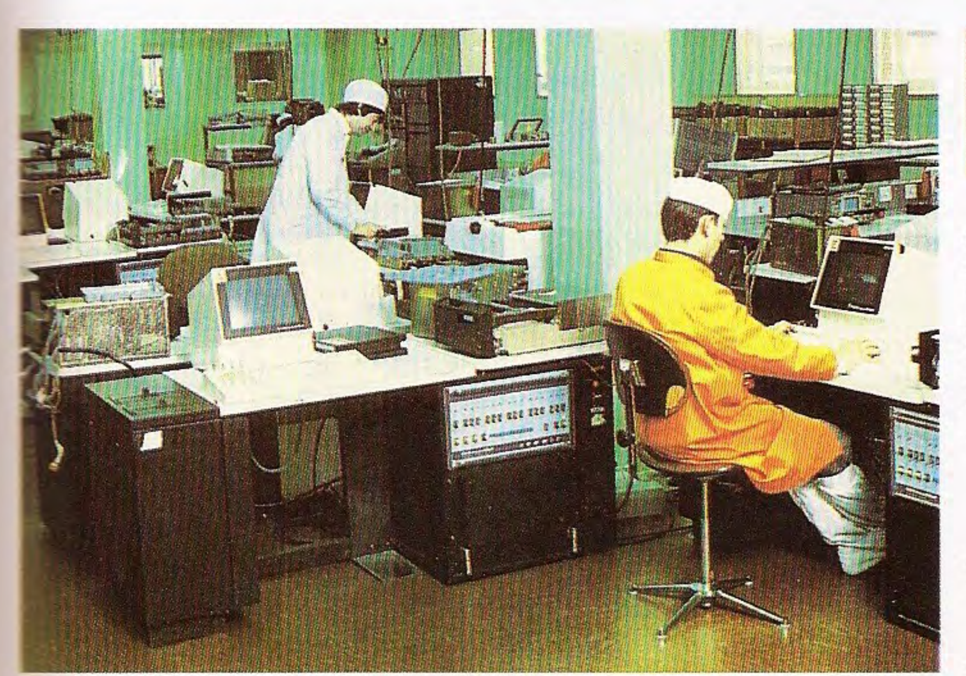




La ecuación $x = a\ e^{bx}$ no se puede resolver por ningún metodo matemático tradicional. Mediante este algoritmo, de tipo iterativo, un ordenador puede calcular el valor de x con un error relativo inferior a ε.



Cuando un número en base diez, con parte fraccionaria se convierte a base dos, aparece un error de truncamiento que afecta a los datos de salida elaborados a partir de este dato.



Los problemas técnico-científicos deben ser analizados y automatizados con especial cuidado para evitar índices de error excesivos en los resultados obtenidos por el ordenador.

evitar errores excesivos en el resultado obtenido por el ordenador. Es por ello importante acompañar al análisis del problema de un estudio del error cometido en el cálculo, de forma que el programa puede indicar si el resultado obtenido es o no válido. El analista debe conocer para ello la teoría de errores.

Procesos de gestión

Los programas de gestión (nómina, contabilidad, control de inventa-



rios, etc.) realizan pocas y muy sencillas operaciones, pero manejan gran cantidad de ficheros y de datos. Ello obliga a prestar especial atención al diseño de los registros y ficheros.

Este tipo de aplicaciones informáticas afectan además a diversos aspectos de una misma empresa e implican, por tanto, al personal de diversos departamentos. El sistema informático interconecta a suministradores de datos y receptores de información de todas estas secciones involucradas en la aplicación.

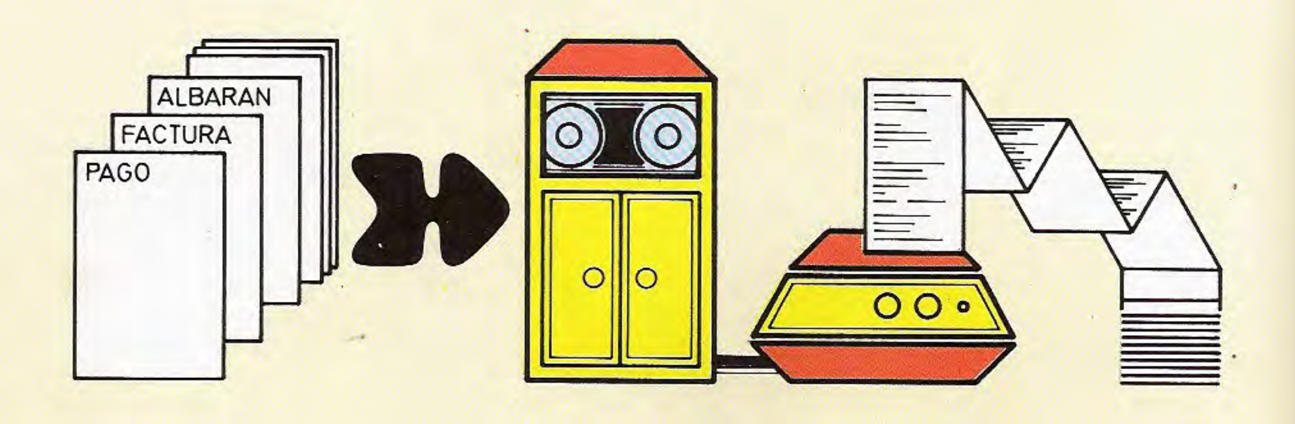
Problemas informatizables

Las diferentes actividades de una empresa, ya sea ésta de tipo familiar o una organización empresarial de gran tamaño, se pueden agrupar en dos grandes tipos:

- Actividades o tareas subjetivas, en las que la intuición desempeña un papel muy importante, como, por ejemplo, la toma de decisiones.

El ordenador es, hoy en día, un elemento omnipresente en la empresa; ya sea automatizando procesos de diseño y fabricación, como solventando las necesidades de gestión.

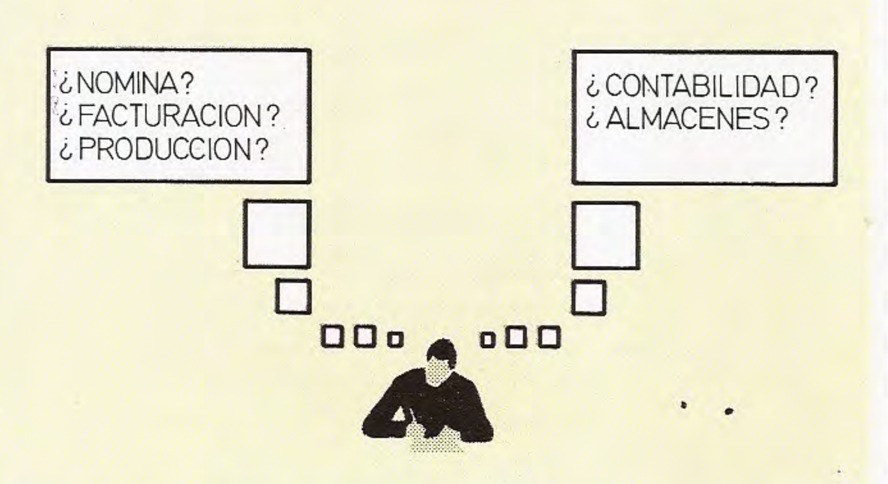
La característica principal de un programa de gestión es el gran volumen de datos y de documentos manejados.



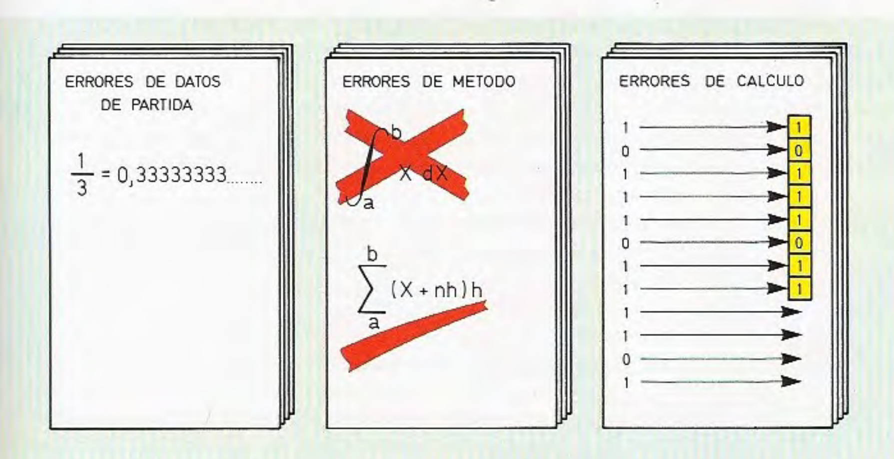
• Actividades o tareas objetivas, que son consecuencia de la aplicación rigurosa de la lógica y la aritmética, como, por ejemplo, el mantenimiento de stocks.

Las tareas objetivas pueden ser, generalmente, mecanizadas. Pero sólo es conveniente informatizarlas cuando el volumen de información a tratar sea grande y deba ejecutarse muy a menudo. Una tarea que trate un gran número de datos y deba realizarse cada cuatro o cinco años, o un cálculo de pocos datos que se realice todos los días, son ejemplos de aplicaciones cuya mecanización puede no ser rentable.

Las actividades subjetivas, por otra parte, no suelen ser susceptibles de mecanización directa. Existe, sin embargo, la posibilidad de utilizar programas de si-



Antes de comenzar la mecanización de un problema de gestión, debe establecerse un plan de prioridades. Es recomendable empezar por los problemas más sencillos y frecuentes, previendo siempre las posibilidades de ampliación del proyecto.



Los valores de salida de un programa de ordenador suelen estar afectados de una serie de errores, la mayoría de ellos inevitabes: errores de redondeo de los datos de partida, error inherente al método de cálculo, y errores por el trucaje de los resultados de las operaciones.

Teoría de errores

En todo cálculo numérico se presentan diversos tipos de errores debidos a los datos de partida, al método empleado en el cálculo y a las limitaciones de la máquina.

Errores en los datos de partida

Un cálculo numérico es un proceso que se realiza a partir de datos de entrada. Los errores de estos datos son errores *inherentes* al proceso, ya que aparecen previamente al cálculo.
Estos errores se deben a diferentes causas, entre las que destacan:

- Las medidas que proporcionan los datos de entrada vienen afectadas de error: es imposible medir exactamente una magnitud física, ya sea una longitud, una masa, un tiempo, etc.
- Los valores de entrada son el resultado de cálculos anteriores afectados por un error; por ejemplo, el dato de entrada es el resultado obtenido tras la ejecución de una subrutina.
- Los valores de entrada son generalmente números decimales que se redondean para introducirlos al ordenador.

Errores de método

En el cálculo numérico muchas operaciones matemáticas son sustituidas por un proceso aproximado.
Así, por ejemplo, el cálculo de una integral se sustituye por el de un sumatorio, por lo que el resultado obtenido es sólo aproximado.

Errores de cálculo

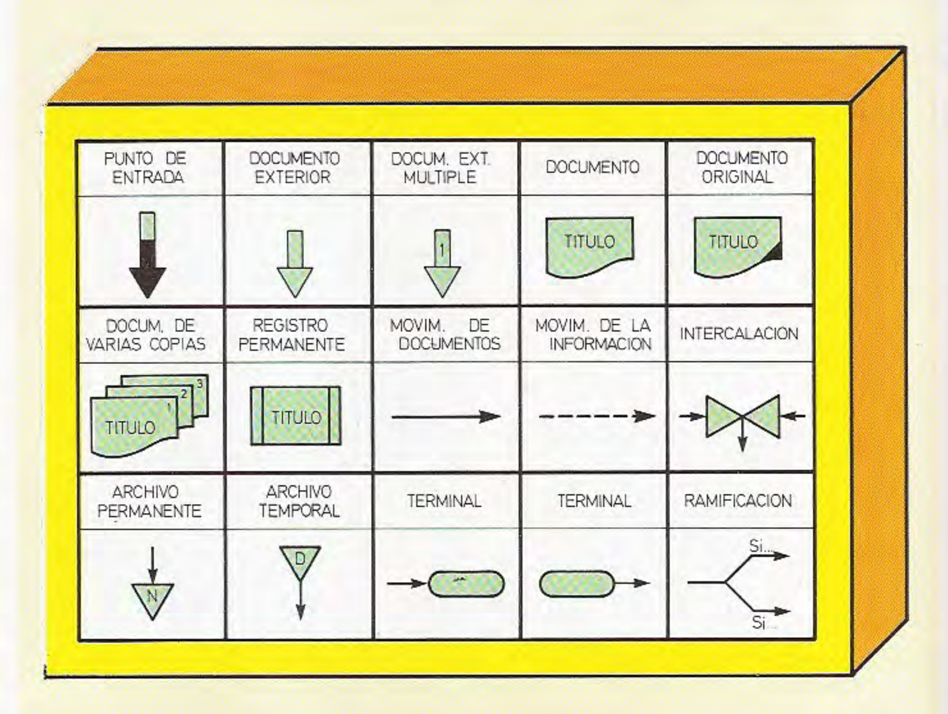
Los cálculos en el ordenador se realizan con números binarios, que tienen un número de cifras limitado. Al trabajar con estos números binarios se produce un tipo de error denominado *round-off:* el resultado de cualquier operación binaria resulta redondeado o truncado.

Organigramas analíticos

El organigrama analítico tiene como principal objetivo facilitar el conocimiento y la comprensión de un proceso informático. El organigrama debe, por ello, reflejar el movimiento de los documentos y el proceso seguido por la información. Debe, asimismo, resaltar los puntos que necesitan más información, indicando la frecuencia con que se manejan los documentos implicados.
Veamos algunos consejos para mejor uso de estos diagramas:
Es importante resaltar el símbolo de creación de un documento (documento original). De esta forma se puede determinar de un vistazo dónde se origina el documento, quién lo creó y cuántas copias del mismo se generan.
Cuando un documento cambia de departamento o punto de proceso, la operación se representa por el símbolo correspondiente.
Los procesos que se llevan a cabo se describen con breves explicaciones colocadas al lado del símbolo.
Los flujos de información se representan por líneas de trazos. Hay que evitar, siempre que se pueda, el cruce de las líneas.
Una planificación adecuada puede eliminar prácticamente el cruce. Si es imprescindible se utiliza «un puente».
Es importante numerar adecuadamente las copias no sólo para poder seguir fácilmente su recorrido, sino también para evitar el cruce de líneas.
Siempre que sea posible, las líneas deben ir de arriba abajo y de izquierda a derecha.
En un organigrama analítico se distinguen cinco zonas principales:

- La primera sirve para especificar los datos del problema y del autor y la fecha de realización.
- Seguidamente se indican los departamentos o personas involucradas en este problema.
- La tercera zona está dedicada al organigrama propiamente dicho.
- En cuarto lugar se anotan los errores detectados sobre la vertical de los puntos donde se hayan localizado.
- Por último, se reserva una zona para anotaciones aclaratorias y explicación de errores.

Los organigramas analíticos son uno de los mejores métodos de representación de los sistemas de información. Permiten, de forma clara y gráfica, una mejor comunicación con el personal no informático implicado en el sistema.



Descripción de todos los símbolos empleados en la realización de un organigrama analítico.

mulación o de cálculo estadístico que disminuyen el riesgo en la toma de decisiones: son los llamados programas de ayuda a la decisión.

Aunque los ordenadores aparecieron para realizar cálculos de tipo matemático, hoy día se aplican sobre todo para la resolución de problemas de gestión. Los constructores de equipos y las empresas de software concentran por ello sus esfuerzos en preparar sistemas que mejoren la rentabilidad de los ordenadores en aplicaciones de gestión empresarial.

Criterios de elección: el análisis previo

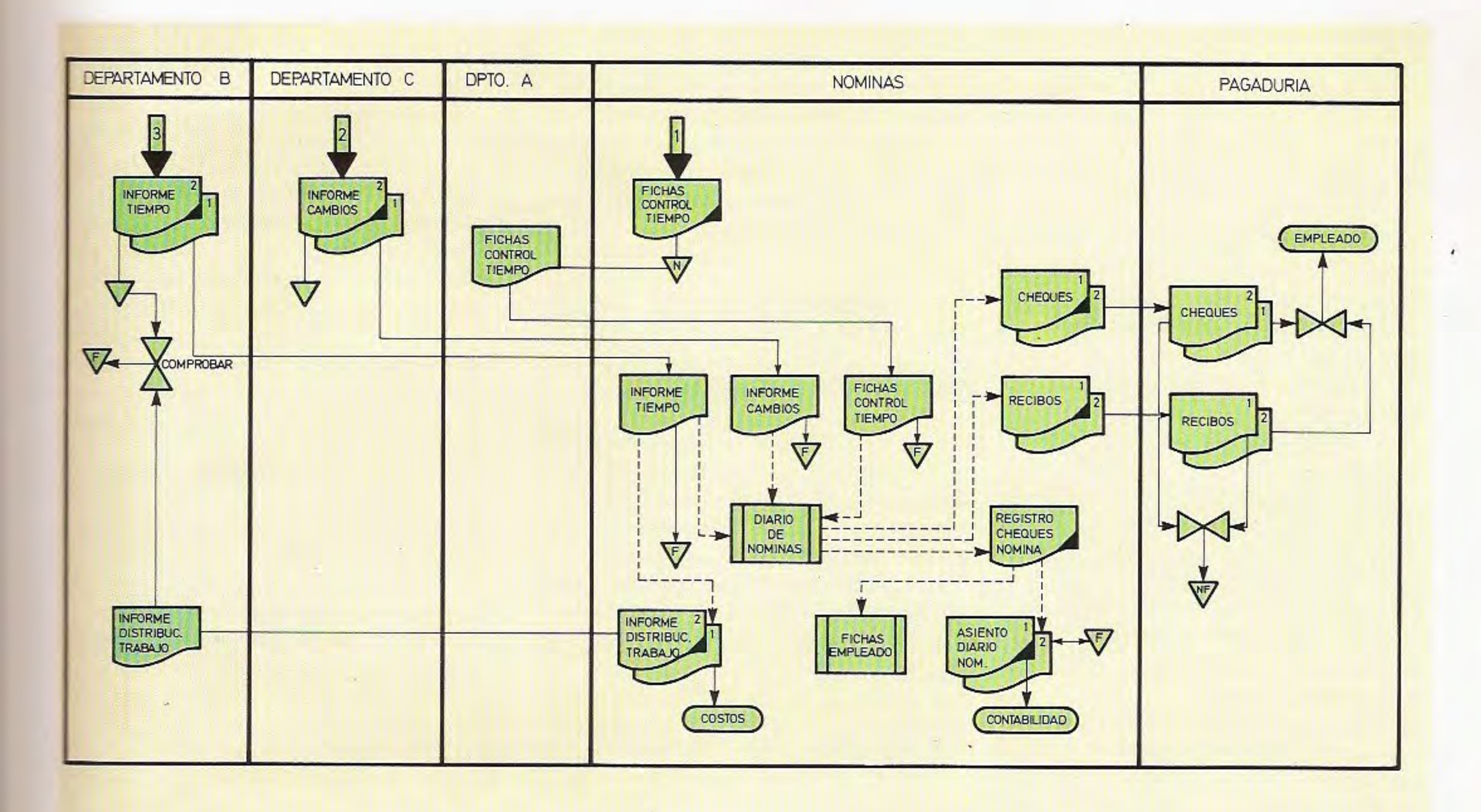
En la fase del análisis previo del proceso se aborda la elección de los problemas o aplicaciones a mecanizar y la elección del ordenador más adecuado para ello.

Los datos que se analizan para decidir qué aplicaciones pueden ser mecanizadas suelen ser los siguientes:

- Volumen de datos a tratar.
- Frecuencia del tratamiento.
- Número de horas-hombre que se emplean para ejecutar la operación manualmente.

Con estos datos, obtenidos para todas las aplicaciones posibles, se procede a elegir las aplicaciones que se van a mecanizar y decidir la prioridad de cada una de éstas. Se siguen, para ello, las siguientes reglas generales:

- Mecanizar primero los problemas más simples, con el fin de pasar a los más complejos en una fase posterior,



Ejemplo de organigrama analítico que describe la realización de la nómina de una empresa cualquiera.

aunque pensando siempre en la mecanización integral de todos los problemas.

- Los problemas más rentables para la empresa o los que se han de realizar en plazos más estrictos, tal como la nómina, tienen prioridad sobre otro tipo de problemas.
- Los problemas estructurados deben mecanizarse antes que los no estructurados.
- No se deben imponer limitaciones *a priori,* ni desestimar la mecanización de aplicaciones sin tener en cuenta las reglas anteriores.

Diagramas del análisis funcional

Durante la fase de análisis funcional el analista intenta llegar a conocer todas las características del sistema y de su entorno para, de esta forma, comprender todo el proceso manual de los datos y poder así acondicionarlo al ordenador.

Debe recabar para ello toda la información posible, generalmente mediante entrevistas con todo el personal afectado por la implementación del nuevo sistema.

La forma más clara de representar esta información es mediante el uso de los organigramas analíticos. Estos organigramas presentan la ventaja de dar una visión rápida del proceso seguido por todos los documentos a través de los diversos departamentos del organismo o empresas.

El organigrama analítico debe de ser aprobado tanto por las personas entrevistadas y consultadas como por los responsables del sistema.

Documentos de entrada e impresos de salida

En las aplicaciones de gestión se necesita usar algún tipo físico (impresos, etc.) de soporte para los datos de entrada al sistema. Es preciso, asimismo, obtener documentos impresos de salida con destino, en la mayoría de los casos, a muchos cientos de miles de usuarios.

El diseño de estos documentos es uno de los trabajos más arduos que debe realizar el analista, si pretende que los usuarios no tengan grandes dudas a la hora de rellenar los cuestionarios de entrada o de intepretar los impresos de salida.

Los informes de salida deben ser fá-

cilmente programados y comprensibles por el destinatario.

Los datos y su administración

La cantidad y diversidad de los datos de una empresa suele gestionarse muchas veces por un administrador de datos. La función de esta figura tiene por objeto la gestión de los datos en sí mismos, independientemente de los procesos informáticos y de los circuitos administrativos por los que pasen. El es quien se encarga de todos los aspectos de la coordinación en el uso de los datos. El administrador no se ocupa, evidentemente, de aquellos datos que se crean y utilizan por un único usuario. La función del administrador de datos puede limitarse a uno de estos niveles:

- Todos los datos de la empresa, estén o no mecanizados.
- Todos los datos mecanizados, tanto en ficheros convencionales o en base de datos.
- Sólo los datos que se encuentran en las bases de datos.

Esta administración de datos comprende funciones de diversos tipos:

- Coordinación: entre aplicaciones, usuarios y trabajos.
- Formación de personal informático y de usuarios.
- Información y documentación.
- Establecimiento de normas de uso, confidencialidad y seguridad.
- Auditoría.
- Diseño.
- Operación.
- Especificación y selección de software de las bases de datos.

Selección del ordenador más adecuado

Para decidir sobre la mecanización de un determinado proceso no basta con conocer el volumen de datos, la frecuencia de tratamiento de éstos y la complejidad de los cálculos aritméticos y lógicos. Se debe encontrar también una solución de compromiso entre la rentabilidad de las aplicaciones a mecanizar y el grado de ocupación del ordenador o, lo que es lo mismo, el coste de alquiler o de amortización del mismo.

Por otra parte, es necesario prever el crecimiento del equipo, que debe ser capaz de adaptarse a posteriores ampliaciones de la mecanización.

Los ficheros y su diseño

Para que la información pueda tratarse mediante un ordenador es necesario agruparla en registros. Cada uno de estos registros debe tener un indicativo que le diferencie de los demás.

Los aspectos más importantes a tener en cuenta en el diseño de ficheros son:

- Longitud de los registros que integran el fichero.
- Número de registros que pueden ser leídos de una vez por el ordenador.
- Complejidad de tratamiento.

Los registros pueden ser de longitud fija o de longitud variable. Cuando el registro es de longitud variable es necesario indicar ésta por un «dato de longitud» colocado al comienzo de cada registro.

La longitud del registro condiciona en parte el tipo de soporte utilizable, ya que algunos de ellos tienen una capacidad limitada.

Un registro que contiene información tratable como una unidad independiente se denomina registro lógico.

Para mejorar el rendimiento de la operación de entrada de un ordenador es necesario agrupar los registros lógicos en los llamados registros físicos. El factor de blocaje, o número de registros lógicos que contiene un registro físico, se determina teniendo en cuenta la velocidad de búsqueda en el soporte elegido, la velocidad de escritura-lectura, la velocidad de búsqueda en memoria, el soporte utilizado, etc.

Los registros físicos también pueden ser de longitud fija o variable.

Cada aplicación determina también el tipo de organización de ficheros y su método de acceso, aunque estas características están también condicionadas por el tipo de memoria de masa conectable al ordenador.



Para saber más

¿Cuáles son los problemas de gestión cuya mecanización resulta más rentable?

Los que, manejando mayor volumen de datos, requieren también una mayor frecuencia de proceso.
La regla general podría ser: «Ni volumen sin frecuencia, ni frecuencia sin volumen».

¿Existe alguna contradicción entre la regla que dice empezar por los problemas más rentables y la que aconseja empezar por los problemas más sencillos?

No, ya que los problemas que se ejecutan con mucha frecuencia y que manejan gran volumen de datos son, generalmente, los que siguen un proceso muy sencillo; por ejemplo, la facturación, el control de almacén, la contabilidad, etc.

¿Se deben mecanizar aplicaciones que por sí mismas no justifican su tratamiento por ordenador?

En algunos casos, sí; por ejemplo, cuando aportan datos a otras aplicaciones rentables o cuando sirven de encadenamiento entre dos tareas mecanizadas.

¿Deben modificarse los impresos de salida para mecanizar un proceso?

La introducción del ordenador en un proceso de gestión supone no sólo efectuar el proceso más rápidamente, sino también una reorganización y optimización del proceso. Puede aprovecharse la ocasión, por tanto, para diseñar un nuevo impreso de salida que, a la vez de ser más claro para el usuario, facilite el trabajo del ordenador.

¿Qué hay que tener en cuenta para calcular el coste de amortización de un ordenador?

Además del coste del equipo, debe tenerse en cuenta los costos de mantenimiento y los gastos del software, tanto básico como de aplicación.

Almacenamiento organizado de la información

Ficheros, registros y campos



a misión más importante de un ordenador es el tratamiento mecanizado de la información, y para ello es fundamental que los datos se organicen de forma que se facilite su gestión. Como ya se ha mencionado a lo largo de la obra, el ordenador almacena la información en base a un elemento unitario llamado bit, que puede valer cero o uno. Si nos quedáramos en este nivel, el ordenador no tendría problemas para realizar la gestión, pero los programadores verían una representación lógica de la información muy alejada de cualquiera de las notaciones a las que están acostumbrados.

En este capítulo se describen las distintas formas de almacenamiento organizado de la información, que permiten no sólo facilitar su interpretación para los programadores, sino también una gestión más eficiente por parte del ordenador.

Registros, campos y ficheros

Llamamos *registro* a un conjunto de informaciones referentes a un mismo tema y que, por tanto, constituyen una unidad de procesamiento.

Dentro de cada registro se puede hacer una división en unidades de tratamiento con entidad propia. A estas unidades se las denomina *campos*.

Cuando la información contenida en un campo es utilizada para la identificación del registro se dice que es un campo indicativo (o clave de registro). De alguna manera se puede decir que una clave es el nombre del registro, de modo que cuando alquien quiere «llamar» a un registro concreto vale con que indique su «nombre» (clave) para poder distinguir al registro deseado de los restantes.

Por último, se llama *fichero* a un conjunto de registros que contienen información sobre un mismo tema.

Los ficheros se distinguen entre sí por su nombre físico. También se les puede asignar un nombre lógico, por el que serán accedidos desde cualquier programa. La forma de ligar el nombre lógico con el nombre físico de un fichero varía según el sistema utilizado.

Los ficheros pueden ser multivolumen cuando se necesita más de un soporte

	NUMERO DE EMPLEADO	APELLIDO	EDAD	SEXO
REGISTRO	4	ROBLES	35	V
REGISTRO	13	FEITO	25	H
REGISTRO	15	LARA	27	V
	⋮	⋮	⋮	⋮
REGISTRO	22	GARCIA	42	H
REGISTRO	25	GOMEZ	33	V
	CAMPO	CAMPO	CAMPO	CAMPO

FICHERO

En la organización de la información se pueden distinguir tres conceptos jerarquizados. Fichero, *que contiene a varios* Registros *que, a su vez, contiene a varios* Campos. *En la figura se representa un posible fichero de personal.*

Sistemas de protección

Para que un usuario no pueda, voluntaria o involuntariamente, destruir informaciones residentes en el ordenador es necesario establecer sistemas de protección, sobre todo cuando se está operando en régimen multiusuario.
A este tipo de protección, indispensable para el buen funcionamiento de los sistemas basados en ordenador, se une a veces la necesidad de confidenciabilidad de ciertas informaciones de algunos usuarios.
Toda información, tanto en memoria principal como en memoria auxiliar, puede tener tres niveles de protección:

1. Nula: podrá ser leída y modificada por cualquier usuario.
2. De escritura: podrá ser leída por cualquier usuario, pero no podrá ser modificada.
3. De lectura: no podrá ser leída ni modificada más que por el usuario a quien pertenezca.

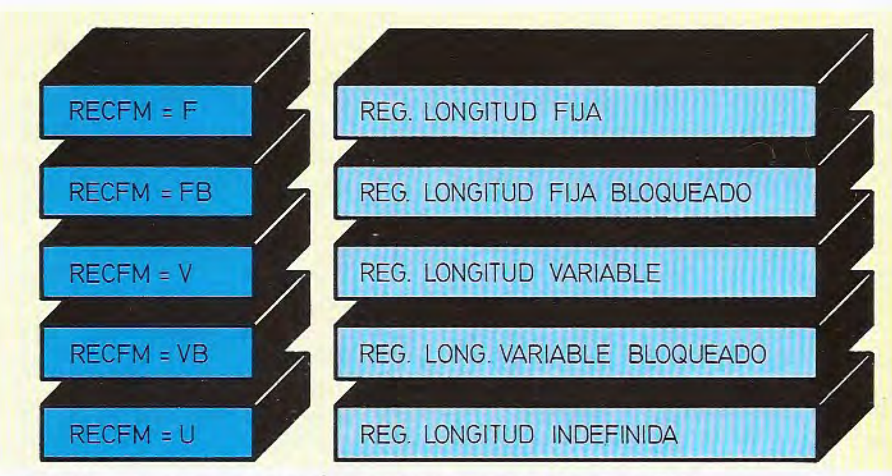
A continuación detallamos la utilidad de la protección de información en memoria central y en memorias auxiliares.

Protección en memoria central

Se limita a prohibir las instrucciones de un programa en ejecución que hagan referencia a zonas de memoria que no le estén autorizadas. Este control se podría realizar mediante un programa que verificara la validez de cada una de las instrucciones del programa controlado, pero el tiempo necesario para realizar este proceso sería tan grande que hace imposible que este método sea rentable. Por tanto, la protección de memoria se realiza, la mayoría de las veces, por medio de dispositivos cableados.

Protección en memoria auxiliar

Utilizando dispositivos físicos es posible proteger algunos soportes de información, haciendo que de ellos sólo se pueda leer (mediante un aro colocado en las cintas magnéticas, por la eliminación de una «pestaña» en los casetes, etc.). Pero la principal protección está encomendada al sistema operativo, que, en consecuencia, deberá supervisar todas las operaciones, tanto de entrada como de salida.



El formato de los registros de un fichero se suele especificar mediante un parámetro llamado RECFM, el cual puede tomar los valores que se indican en esta figura.



Además de con el parámetro RECFM, las características generales de los ficheros suelen especificarse mediante los parámetros que se indican en esta figura.

de información para su almacenamiento, o, en caso contrario, los soportes se llaman multifichero cuando disponen de suficiente espacio para almacenar más de un fichero.

Toda la información referente a las características generales de un fichero se suele almacenar al principio de la zona ocupada por los datos que contiene; dicha zona se denomina etiqueta del fichero.

Formato de registros y campos

Los campos contenidos en un registro pueden ser de longitud fija o variable y, análogamente, los registros de un fichero pueden también ser de longitud fija o variable.

La gestión de los tamaños, tanto de los campos como de los registros de longitud variable, es responsabilidad de los programas de usuario. Evidentemente es más fácil realizar un programa que trate ficheros con todos los registros del mismo tamaño y que a su vez estén compuestos por campos del mismo tamaño; pero a veces la propia naturaleza de los datos obliga a adoptar ficheros más flexibles.

Para optimizar el proceso de intercambio de información es necesario minimi-

zar el número de acceso a las unidades de entrada/salida, y para ello es conveniente agrupar varios registros lógicos en un único registro físico o bloque, que será la unidad de información que se transfiere en cada orden de E/S.

Tipos de ficheros

Se pueden realizar muchas clasificaciones entre los ficheros según se consideren unas características u otras. Las principales son las siguientes:

1. *Tamaño*

Según sean las longitudes de los registros y el número de registros que previsiblemente contendrá el fichero, se puede calcular su tamaño.

2. *Volatilidad*

Según el número de altas y bajas que se vayan a producir a lo largo del tiempo, se puede estimar los «movimientos» que tendrá el fichero.

Por ejemplo, un fichero de personal de una empresa a la que cada mes se incorporen muchos empleados y se eliminen también muchos tendrá un elevado número de movimientos y, por tanto, será recomendable utilizar algún dispositivo de acceso directo. En cambio, para un fichero de departamentos en el que se especifiquen los datos de cada uno de ellos en un registro, salvo reorganizaciones empresariales, no tendrá prácticamente ningún movimiento, por lo que se podrá mantener perfectamente en dispositivos secuenciales.

3. *Actividad*

La frecuencia con que es referenciado un fichero nos mide la actividad que tiene y ayuda a elegir el dispositivo físico.

Si continuamos con la empresa de los ejemplos anteriores y con su fichero de departamentos, el hecho de que dicho fichero pueda ser secuencial debido al bajo número de movimientos que tiene puede implicar la utilización de una cinta magnética para su almacenamiento. En cambio, si dicho fichero tiene mucha actividad, es decir, referenciado desde muchos programas, será interesante mantenerlo en un soporte permanente de acceso directo (generalmente en un disco magnético).

RECFM	AREA DE DATOS	OTROS PARAMETROS
F	BLOQUE = REGISTRO	BLKSIZE
FB	REG. - 1 | REG. - 2 | … | REG. - M (BLOQUE)	LRECL BLKSIZE = M. LRECL
V	LONG. DEL BLOQUE | BLOQUE = REGISTRO	BLKSIZE = TAMAÑO MAX. + 4
VB	← REG. → ← REG. → ← REG. → LONG. BLOQUE, DATOS, LONG. REG.	BLKSIZE Y LRECL BLKSIZE = 4 + M. LRECL LRECL = 4 + max. tamaño registro logico
U	BLOQUE = REGISTRO	BLKSIZE = MAX. LONG. DE UN BLOQUE

En la figura pueden observarse los parámetros necesarios para definir el área de datos en un registro, en función de los valores del parámetro RECFM.

Se pueden distinguir dos tipos de ficheros según su actividad:

- *Ficheros permanentes*

Son los que se encuentran continuamente a disposición de los programas que los referencien.

- *Ficheros de trabajo*

Son creados por un programa de forma temporal. Cuando su utilización ha terminado son borrados.

Seguridad y control

En algunos casos la pérdida de la información almacenada en un fichero no tiene ninguna trascendencia, bien porque los datos ya no eran necesarios o bien porque se puedan reconstruir fácilmente. Sin embargo, en otras ocasiones la información perdida puede tener plena vigencia y su reconstrucción puede resultar muy cara o complicada.

Existen básicamente dos motivos para aumentar la seguridad y el control sobre los ficheros almacenados:

1. Pérdida o deterioro de la información almacenada en el fichero.
2. Acceso e información confidencial por usuarios no autorizados.

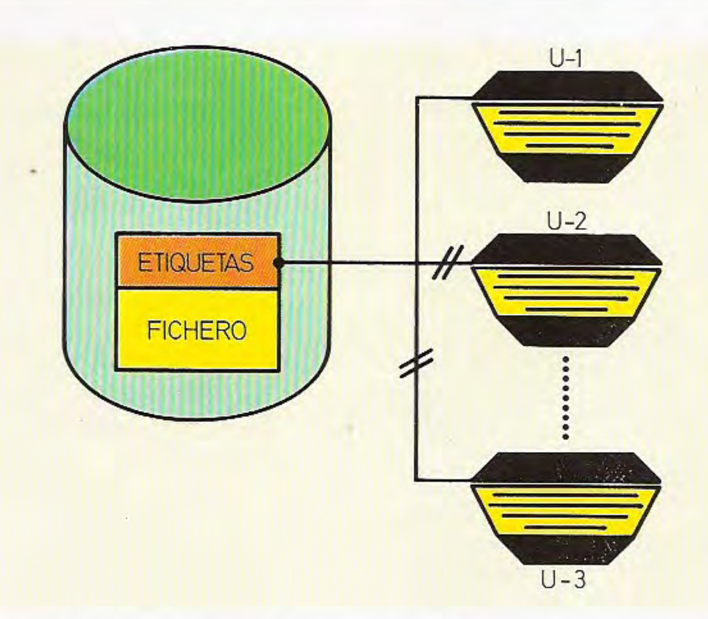
La forma más normal de resolver el primer problema es usando copias de seguridad del fichero. Estas copias se realizan cada vez que se modifican los datos, y así, si se pierde la información, se podrá recuperar el fichero en su estado final.

Para resolver el segundo problema, es decir, para garantizar que los datos contenidos en un fichero confidencial no son accedidos por personas no autorizadas, cada fichero está identificado en el sistema mediante su etiqueta, que contiene, además del nombre del fichero, las fechas de creación y expiración, así como información de si el fichero está protegido contra la lectura y/o escritura.

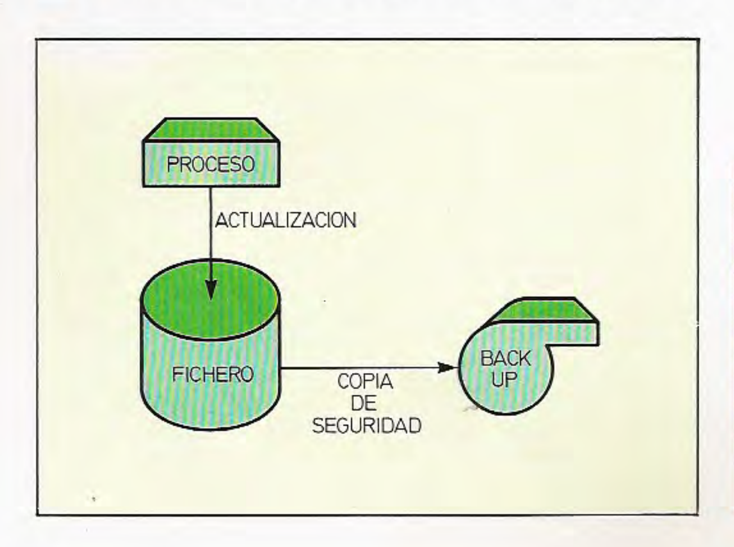
Organización de ficheros

Para terminar este capítulo veremos algunos tipos de organización de la in-

PRINCIPALES OPERACIONES SOBRE FICHEROS
CREACION Se realiza con la primera carga. CONSULTA Y ACTUALIZACION Consiste en la lectura de la información para facilitar la consulta del usuario y la escritura para la modificación, inclusión o supresión de partes del fichero. CLASIFICACION Y FUSION Ordenación de los registros según un determinado criterio (de uno o más ficheros). REORGANIZACION Sólo para ficheros indexados.



En las etiquetas de un fichero se incluyen sus principales características, así como los datos necesarios para garantizar que dicho fichero es sólo utilizado por usuarios autorizados. En la figura sólo está autorizado el usuario 1.



Cada vez que se actualiza un fichero maestro es muy conveniente hacer una copia de seguridad para garantizar que el posible deterioro del fichero original no suponga la pérdida de toda la información.

formación en ficheros. A la hora de elegir el tipo de organización óptima habrá que tener en cuenta los siguientes cuatro puntos:

1. El tiempo y forma de acceso deseados para un registro genérico del fichero.
2. Procedimiento utilizado para insertar nuevos registros en un fichero ya creado.
3. Espacio ocupado o, lo que es equivalente, coste de mantenimiento.
4. Tipo de procesamiento de la información del fichero. Las organizaciones serán muy distintas según se utilicen programas *batch* u *on-line*.

Dentro de las posibles organizaciones podemos distinguir:

1. Organización secuencial. Cada registro se almacena en la posición siguiente a la anterior.
2. Organización secuencial indexada. Se distinguen dos conjuntos de datos: uno, denominado área primaria, en el que se encuentra la información, y otro, denominado área de índices, que permite acceder rápidamente a los registros del área primaria.
3. Organización secuencial encadenada. Cada ítem de información tiene un enlace con el siguiente. Para añadir nuevos ítems es necesario romper el enlace antiguo.
4. Organización particionada. Es una variante de la organización secuencial, con la salvedad de que el fichero puede estar dividido en dos partes llamadas miembros.
5. Organización aleatoria o directa. Permite acceder directamente al registro cuya clave coincida con la clave solicitada.

Para saber más

¿Cuáles son los principales conceptos necesarios para la organización lógica de la información?

El concepto más importante es el *fichero,* el cual está constituido por un determinado número de *registros,* que a su vez están formados por datos elementales llamados *campos.*

Bases de datos

Características y modelos organizativos



En la actualidad las empresas tienen que utilizar un gran número de aplicaciones mecanizadas y, por consiguiente, crece también el número de ficheros necesarios para tratar estas aplicaciones. Muchas de las aplicaciones pueden necesitar tratar un dato de un determinado fichero de forma simultánea, de manera que son pocos los ficheros de uso exclusivo para una determinada aplicación. Lo normal es que un mismo fichero tenga que ser utilizado de diferentes maneras por distintas aplicaciones.

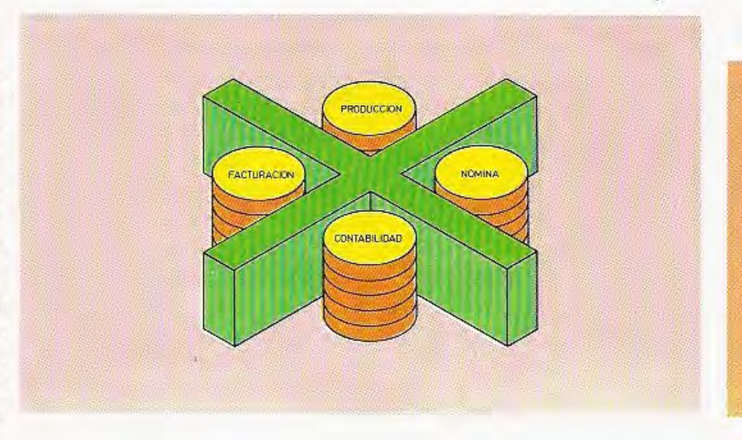
Suponga que se tiene un archivo de pedidos de material de una empresa. Este archivo se puede utilizar para varios trabajos, como, por ejemplo, ver el volumen de ventas de la empresa, observar si hace falta suministrar más material al almacén de pedidos, ver cuáles son los materiales que se venden en mayor cantidad, actualizar el archivo, etc.

En este ejemplo hay datos que pueden ser empleados en varias aplicaciones o trabajos.

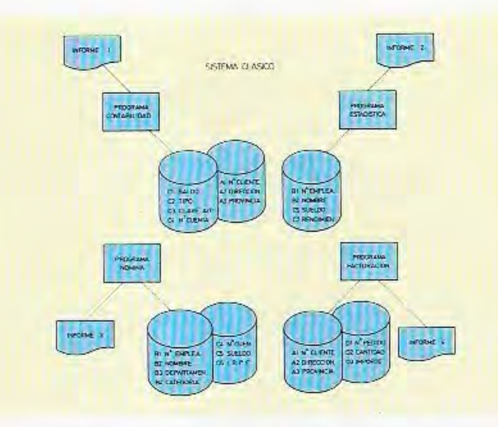
Para evitar que existan datos repetidos en distintos ficheros, lo que se hace es almacenar un dato una sola vez, en un fichero accesible a las distintas aplicaciones. Esta es la idea que culminó en la creación de la *base de datos,* que se puede definir como «un almacenamiento centralizado de datos relacionados entre sí, que pueden utilizarse en distintas aplicaciones».

Orígenes históricos

Para comprender el nacimiento y evolución de los sistemas de bases de datos es necesario conocer el medio informático de los últimos años. Las aplicaciones que necesitaban de la utilización de ficheros de datos operaban únicamente con sus propios ficheros. El único caso en el que varias aplicaciones utilizaban ficheros comunes era cuando una de ellas creaba el fichero y lo accedía para editar los informes necesarios y, posteriormente y de forma independiente, una segunda aplicación utilizaba el fichero para otras labores.



En una aplicación que no utiliza base de datos, cada programa emplea sus propios archivos, independientes de los demás, aunque contenga datos comunes a varios de ellos.



En un sistema tradicional de tratamiento de datos cada programa de aplicación utiliza sus propios ficheros, y no tiene acceso a las informaciones que manejan los restantes programas.

Los sistemas funcionaban sin multiprogramación, es decir, el ordenador ejecutaba simultáneamente un único programa y, por tanto, la seguridad e integridad de los datos estaban garantizadas con tal que los programas realizaran las operaciones correctamente.

El concepto de «integridad de los datos» puede ser entendido de diferentes maneras. En este caso se entiende como integridad de los datos el que éstos no puedan ser borrados por error, y que si por circunstancias ajenas al sistema se deteriora un fichero, éste se pueda regenerar con la información que contenía antes de su destrucción de forma cómoda y rápida.

A su vez, el concepto «seguridad de los datos» hace referencia a que los datos estén protegidos contra modificaciones no autorizadas, tanto altas como bajas o cambios, consiguiendo de esta forma evitar malas operaciones que pudieran realizarse accidental o intencionadamente.

En los primeros años del proceso de datos, la seguridad se conseguía mediante procedimientos drásticos. Por ejemplo, si una cinta magnética contenía una zona de información confidencial, su propietario o usuario se encargaba de guardar personalmente dicha cinta, con lo que si por cualquier motivo no se encontraba presente en la empresa, el resto de la organización no podía acceder a subconjuntos de la información no confidencial que pudieran necesitar para otras aplicaciones.

De alguna manera los programadores eran los auténticos dueños no sólo de los programas, sino también de los ficheros de datos.

Al ampliarse las aplicaciones y desarrollarse otras nuevas muy ligadas a los mismos ficheros utilizados por las aplicaciones antiguas, y al implementarse sistemas operativos más potentes que permitían la multiprogramación (es decir, que en un momento dado se podían ejecutar distintos programas que actuaban contra los mismos ficheros), aumentaron las dificultades tanto en la integridad de los datos como en la seguridad de la información.

Redundancia e inconsistencia

Se dice que un dato es redundante cuando se encuentra duplicado en más de un fichero.

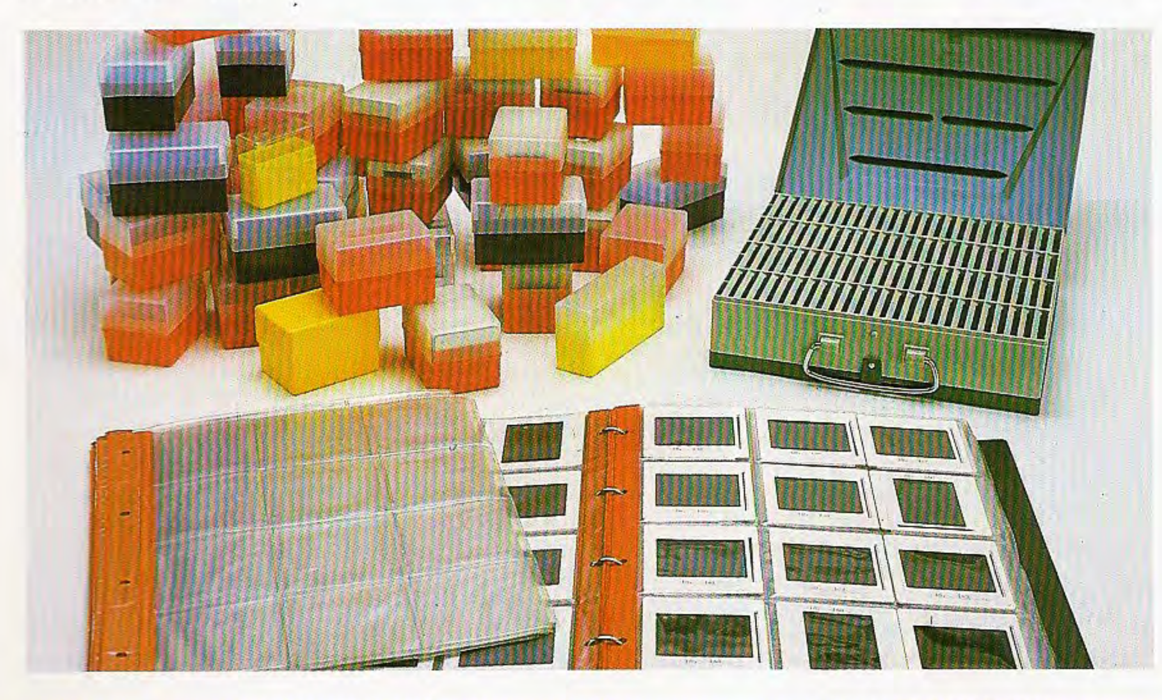
Cuando los sistemas operativos fueron relativamente potentes y se empezaron a utilizar dispositivos de acceso directo, a medida que se iban desarrollando nuevas aplicaciones se creaban nuevos ficheros, sin comprobar que algunos de los datos de estos ficheros existían en otros creados anteriormente. Dicho de otra forma: la redundancia crecía vertiginosamente.

Esta duplicidad de datos suponía una necesidad de más espacio en disco que el estrictamente necesario y, por otra parte, cuando era necesario actualizar un dato que se encontraba en más de un fichero, se debía repetir el proceso de actualización una vez para cada fichero, lo que se traducía en un tiempo adicional, tanto de proceso como de programación, y nuevas oportunidades para cometer errores.

De la redundancia se deduce casi siempre la inconsistencia, es decir: dos o más datos redundantes tienen distintos valores en un mismo momento.

Por ejemplo, suponga que un director solicita un informe de la situación del personal a dos departamentos distintos (podrían ser contabilidad y el propio departamento de personal), que mantienen ficheros independientes y cada uno de ellos refleja una situación distinta. La pregunta sería: ¿De cuál de los dos informes me fío? Y la contestación sería tajante: ¡De ninguno de los dos! La forma lógica de reducir esta inconsistencia es unificar los ficheros existentes que contienen los datos duplicados.

Mediante esta reunión se elimina el problema de la redundancia y la incon-



Las bases de datos permiten proyectar el concepto de almacenamiento y gestión organizada a grandes volúmenes de información.

FICHERO DE FACTURACION

CLIENTE	PRODUCTO	UNIDADES VENDIDAS	PRECIO PRODUCTO
1	1	7	67
1	3	8	93
2	1	13	67
3	2	9	68
3	3	43	93
3	1	5	67

FICHERO DE PRODUCTOS

PRODUCTO	PRECIO PRODUCTO
1	67
2	68
3	93

En una organización que mantenga los ficheros de facturaciones y productos de la figura, se produce una redundancia en el dato «Precio del producto», ya que se encuentra almacenado en ambos ficheros.

sistencia, pero potencialmente pueden surgir otros nuevos problemas, entre los que destaca uno: la protección de los datos debe aumentarse, ya que usuarios distintos deben acceder al mismo fichero.

Independencia de los datos

Para evitar la necesidad de que al modificar la estructura de almacenamiento de los datos sea necesario alterar los programas que se ocupan de su tratamiento, es preciso que los referidos programas sean independientes de los datos a procesar. Esto significa que las variaciones en el formato de los registros en la organización de los ficheros o en los dispositivos de almacenamiento no afecten a los programas que los utilizan y, por tanto, no exijan modificaciones en los mismos.

Para implementar esta independencia entre programas y datos es necesario analizar con más detalle el concepto de información y los métodos utilizados para representarla.

La información consiste en acumulación de ideas y hechos acerca de diferentes cosas: personas, lugares, herramientas, etc. Estas cosas se denominan entidades. Es decir, se registra información sobre entidades.

La información sobre las entidades consta de tres partes:

1. Contexto, que define el ámbito de la identidad. Es decir, el contexto es el mismo si las entidades son iguales y diferente si son distintas.
2. Datos, que marcan unos valores concretos para las variables del ámbito de la identidad.
3. Representación de los datos, que

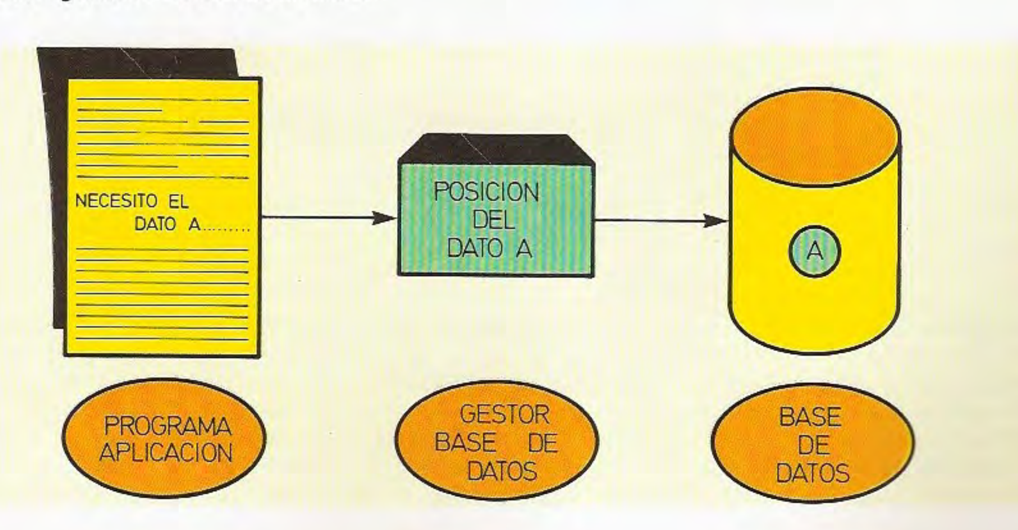
FICHERO DE FACTURACION

CLIENTE	PRODUCTO	UNIDADES VENDIDAS	PRECIO PRODUCTO
1	1	7	65
1	3	8	93
2	1	13	65
3	2	9	68
3	3	43	93
3	1	5	65

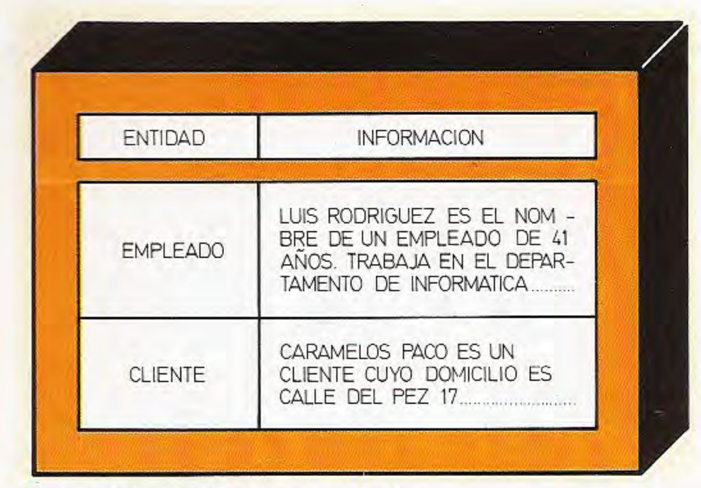
FICHERO DE PRODUCTOS

PRODUCTO	PRECIO PRODUCTO
1	67
2	68
3	93

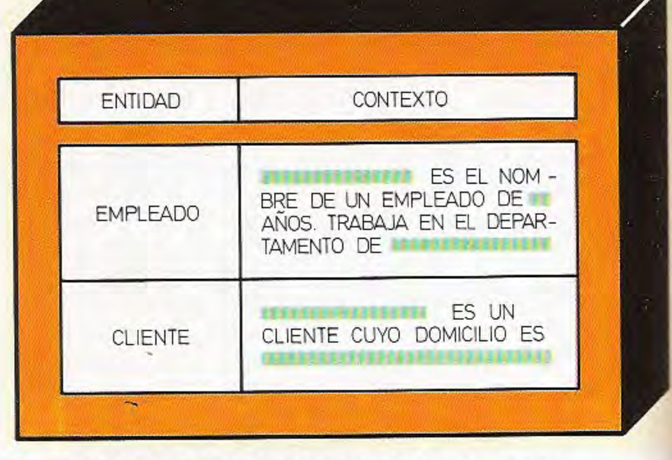
En la figura se muestra un ejemplo de inconsistencia: el precio del producto 1 tiene diferente valor en cada uno de los ficheros... ¿Cuál de los dos es el bueno?



La independencia entre datos y programas se consigue en las bases de datos mediante el gestor. Aunque se cambie la estructura de la información en la base de datos, basta con notificarlo al gestor y el programa no tendrá por qué ser modificado.



La «información» se corresponde con lo que se conoce de una determinada entidad.



Una de las zonas constitutivas de la información sobre una entidad es el «contexto» donde se define el ámbito de la misma.



...La otra zona coincide con los datos propiamente dichos.

indica el formato físico con que se representan los datos.

Bases de datos

Ahora ya estamos en disposición de distinguir entre sistemas tradicionales de ficheros y bases de datos. Tradicionalmente los programas utilizaban un registro lógico para las operaciones sobre ficheros; era, pues, el programa el que debía conocer la secuencia de campos en el registro lógico y los formatos de registro. Así pues, si por cualquier razón era necesario modificar la estructura del registro lógico, había que modificar también el programa.

En un sistema de bases de datos, la situación es muy diferente. Las principales innovaciones aportadas son las siguientes:

1. En lugar de existir muchos ficheros independientes, dirigidos cada uno a una aplicación, existe una única base de datos, que se puede definir como un conjunto de datos no redundantes y relacionados, que pueden ser procesados por una o más aplicaciones a la vez.

2. Las lecturas o escrituras en la base de datos no se realizan directamente desde los programas de aplicación, sino que se realizan mediante el sistema de gestión de la base. Por tanto, si se modifica la estructura de la base, no es necesario modificar todos los programas que acceden a ella.

3. El sistema de gestión de la base de datos utiliza dos tipos de bloques de control almacenados en disco:

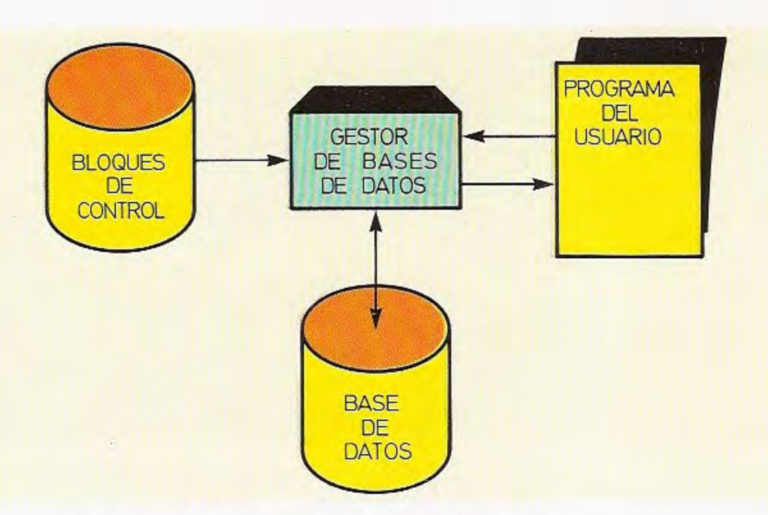
- Unos describen las características físicas de la base de datos.
- Otros especifican el subconjunto de datos con los que puede trabajar un determinado programa de aplicación.

Características esenciales de una base de datos

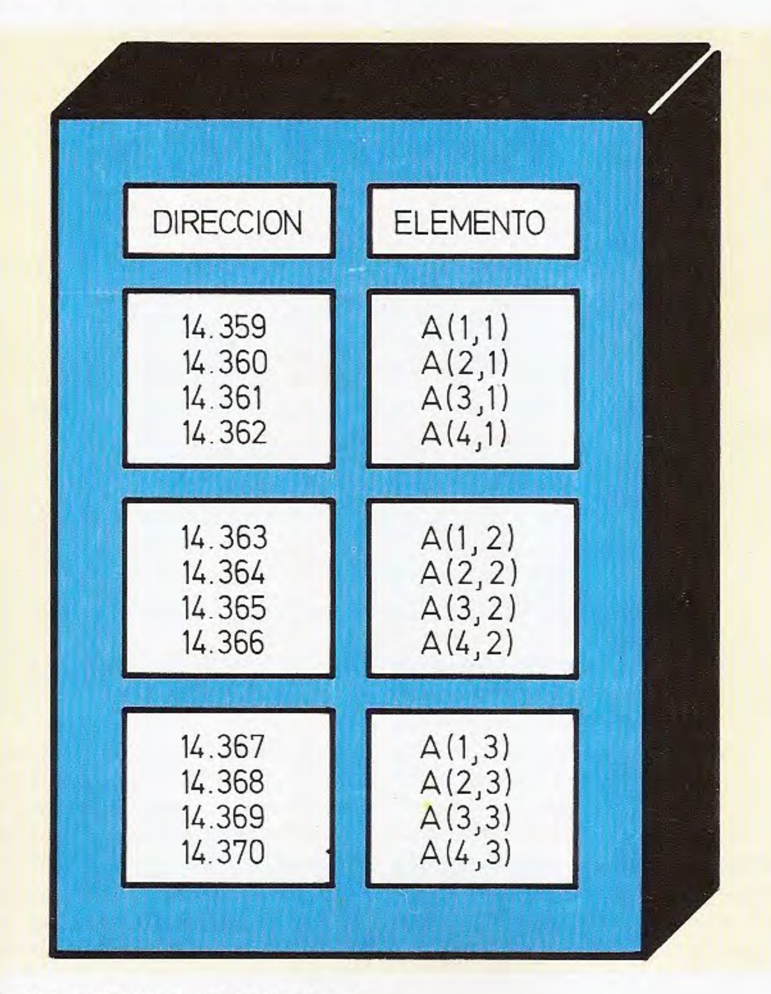
Para que una base de datos pueda cumplir su función de una forma adecuada es necesario mantenerla, poder acceder a sus datos fácilmente, relacionar los datos entre sí, etc. De ahí que sea preciso contar con un sistema de tratamiento que organice de forma adecuada los ficheros, además de suministrar un lenguaje apropiado para acceder a los datos y los programas necesarios para el mantenimiento de los ficheros.

Una base de datos debe reunir una serie de características esenciales; las más importante son las que se relacionan a continuación:

- Los datos y las relaciones entre ellos se deben determinar y organizar mediante un modelo de representación que encuadre las estructuras jerárquicas y relacionales.
- Puede ocurrir que las diferentes aplicaciones que va a utilizar el banco



El gestor de bases de datos lee información de los bloques de control y actúa sobre la base de datos según las órdenes que recibe del programa de usuario.



Almacenamiento en memoria de una matriz de dimensión 4 x 3 con la dirección de base 14.359.

de datos traten partes aisladas de los ficheros, por lo que debe existir la posibilidad de describir los datos del fichero desde el punto de vista físico (es decir, como están los datos realmente grabados en el fichero) y desde el punto de vista lógico o por la forma en que ve una determinada aplicación a los datos en el fichero.

- Esta descripción lógica de los datos en una aplicación no tiene que verse

Gestión de arrays en memoria

Cuando el compilador traduce un programa fuente a programa objeto, transforma todos los nombres simbólicos en direcciones de memoria. Esta traducción consiste en transformar los identificadores de variables en números.
Si la variable traducida es un ítem elemental, el resultado de la transformación es también elemental; en cambio, cuando se trata de un elemento de un array, la traducción se convierte en un par de números para identificar la dirección en que se encuentra almacenado. El primero (b) sirve de base y es común para todos los elementos del array; en cambio, el segundo (d) indica el desplazamiento sobre la base para cada elemento concreto.
En el caso de arrays de una única dimensión, la dirección del array vendrá dada por la suma de la base común b más el desplazamiento propio del elemento; así, la dirección del elemento i-ésimo del vector d será igual a i — 1 más la base.
Cuando el array tiene más de una dimensión, el cálculo del desplazamiento se hace por orden decreciente dentro de la secuencia de índices. Es decir, supuesto un array A con dos dimensiones, D_1 y D_2, el elemento A(i,j) se direccionaría sumando a la base b el siguiente desplazamiento:

$D = (j - 1) \cdot D_2 + (i - 1)$

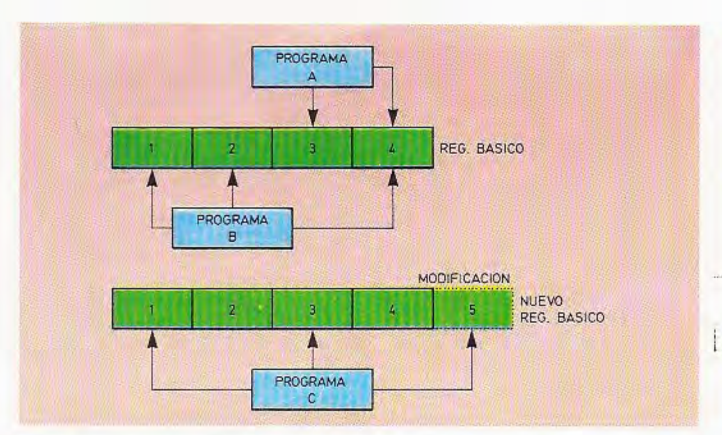
Para aclarar este concepto veamos un sencillo ejemplo:

Supongamos que $D_1 = 4$, $D_2 = 3$ y la base de desplazamiento b = 14.359.
¿En qué dirección de memoria se encuentra almacenado el elemento A(3,2)?

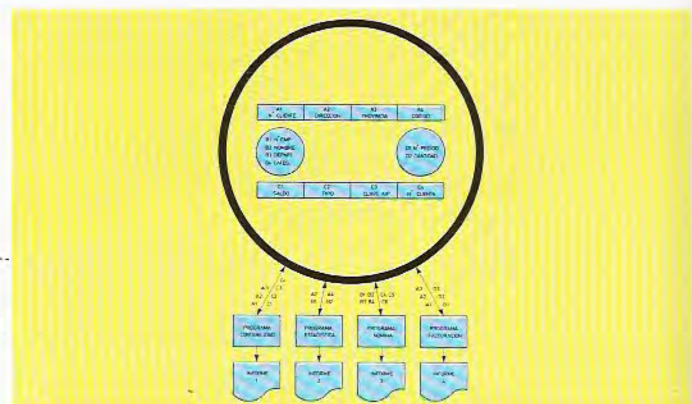
$d = (2 - 1) \cdot 3 + (3 - 1) = 3 + 2 = 5$

Luego la dirección real (r) será:

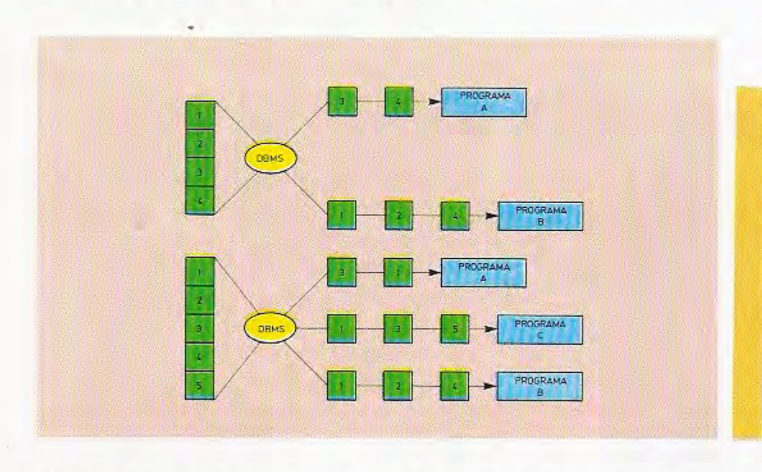
r = 14.359 + 5 = 14.364



Cualquier modificación de un fichero en un sistema tradicional obliga a la modificación de todos los programas cuya ejecución implica a tal fichero.



En un sistema de base de datos las informaciones están centralizadas: sólo existe un fichero a cuyos datos acceden varios programas distintos.



Un cambio en un registro de una base de datos nos obliga a variaciones de los programas en uso, ya que el gestor de la Base (DBMS) entrega a cada programa sólo los datos que necesita.

afectada por adiciones de nuevos datos a los ficheros de la base de datos, ni tampoco por la modificación del soporte o del dispositivo sobre el que se graban los datos para formar los ficheros. A esto es a lo que se llama independencia de datos e independencia del dispositivo.

- Una base de datos constituye, generalmente, una colección completa de datos a la que se puede acceder de diferentes formas, según los distintos tipos de aplicaciones, por lo que es de gran importancia tener mecanismos de seguridad en el acceso a dichos datos. Pueden existir datos confidenciales no accesibles por todos los usuarios.

- Como cada dato debe estar representado una sola vez en una base de datos, es necesario disponer de funciones que aseguren la integridad de los datos. Para ello, cuando se modifique algún dato, lo cual puede afectar a modificar sus relaciones con otros datos, el sistema debe comprobar de forma automática si esa modificación ha quedado correctamente realizada y, en caso contrario, deshacer los cambios o modificaciones realizadas.

Se debe proteger al dato o datos que están siendo accedidos por un programa, aunque en ese momento se encuentre trabajando otro programa en el ordenador que intente acceder a los mismos datos. Esta protección no es necesaria cuando ambos programas accedan a los datos con la única intención de leerlos.

- Es necesario que la base de datos tenga procedimientos y programas que permitan hacer una recuperación de la base de datos en caso de destrucción total o parcial de la misma, así como poner de nuevo en funcionamiento el sistema, una vez que se haya recuperado la base de datos.

Deben existir también programas para la reorganización de la base de datos cuando, por ejemplo, exista una gran cantidad de datos que se quieren incorporar a la base, o bien cuando se quieran cambiar las relaciones entre los datos existentes en la base.

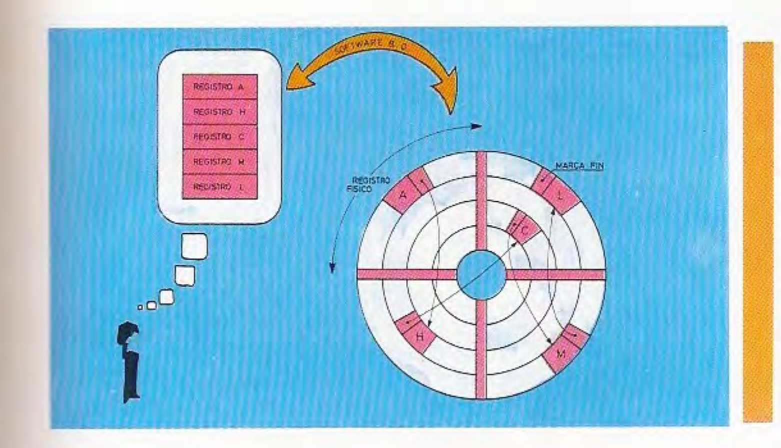
Modelos de la base de datos

Existen diferentes modelos de representación de los datos contenidos en una base de datos y diferentes relaciones entre ellos, es decir, cómo un dato tiene que estar relacionado con otros, independientemente de cómo estén grabados físicamente. Existen tres estructuras fundamentales de organización: jerárquica, relacional y de red.

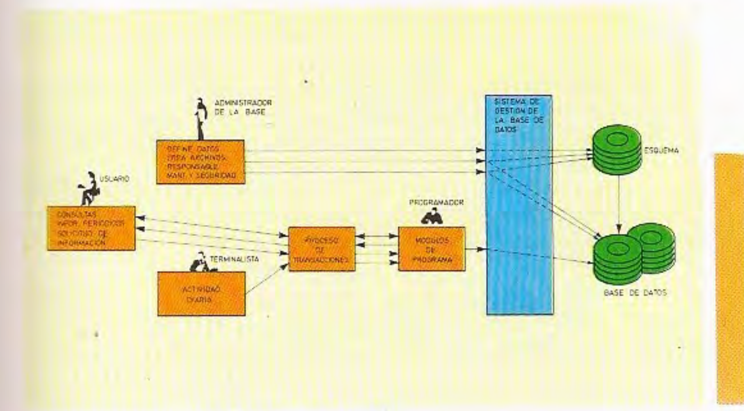
Modelo jerárquico

Los datos se organizan de acuerdo a una jerarquía, de manera que cada elemento dependa solamente de otro anterior o por encima de él y que lo comprenda. Esto es válido para todos los elementos de la estructura, excepto para uno de ellos que no depende de ninguno, llamado *raíz,* y del que dependen todos los demás.

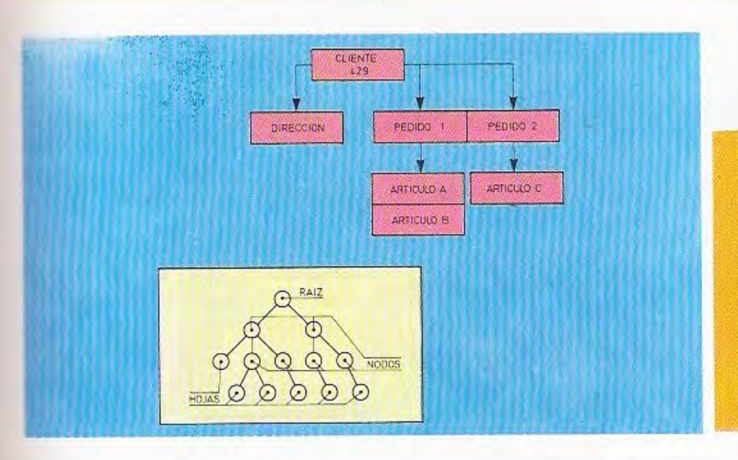
Este modelo se explica en la correspondiente figura. El cliente 429 tiene una dirección posible y, en este momen-



El programa que utiliza una base de datos sólo necesita conocer la distribución lógica del registro. El software de la base relaciona la distribución lógica de los datos con su distribución física.



Modelo conceptual de Base de Datos. Su sistema de gestión se encarga de la coordinación de la base con las diferentes aplicaciones.



Estructura jerárquica de base de datos. Del segmento cliente 429 dependen los segmentos dirección y pedido. Pedido 1 abarca, a su vez, a los artículos A y B.

to, tiene pendientes dos pedidos. El pedido 1 abarca los artículos A y B, y el pedido 2, el artículo C. Cada elemento distinto que compone el modelo recibe el nombre de *segmento*. Nuestro ejemplo tiene cuatro segmentos: cliente, dirección, pedido y artículo. Cada segmento puede tener, a su vez, distintos valores que se llaman *ocurrencias* del segmento. El segmento dirección tiene una ocurrencia: el pedido 2.

Cuando se graba este modelo sobre un dispositivo físico, debe preverse alguna forma de indicar la relación entre las diferentes ocurrencias de los segmentos; es decir, pedido 1 relacionado con artículos A y B. Esta relación se mantiene bien por una continuidad física o por unos bytes de unión llamados *apuntadores*.

Dentro de los modelos jerárquicos de sistemas de bases de datos, uno de los más extendidos es el IMS de IBM.

Este sistema se ha diseñado para fa-

Para saber más

¿En qué consiste la redundancia?

Es el almacenamiento de un mismo dato dos o más veces, bien porque sea necesario para dos aplicaciones y cada una de ellas utilice ficheros independientes, o bien porque los ficheros de una misma aplicación estén mal diseñados.

¿A qué se llama inconsistencia?

Se dice que un dato redundante es inconsistente cuando en un mismo momento el dato tiene diferentes valores en cada uno de los ficheros en los que se encuentre almacenado.

¿Cómo se eliminan la redundancia y la inconsistencia?

En los sistemas tradicionales de ficheros, produciendo un único fichero que contenga todos los datos. Esta unión incorpora nuevos problemas, por lo que una solución más aceptable consiste en la utilización de una base de datos.

¿A qué se llama independencia de los datos?

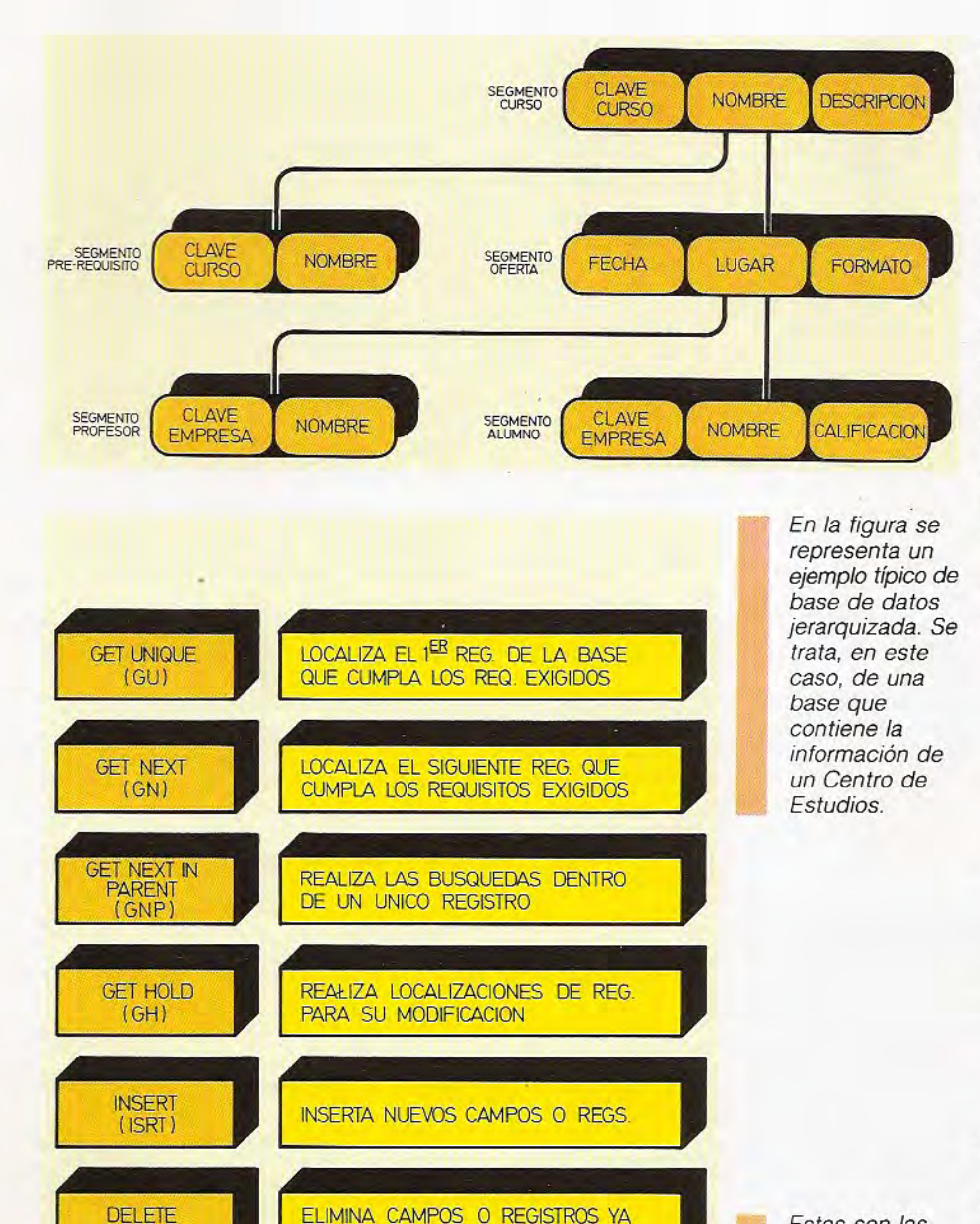
Se dice que un programa es independiente de los datos cuando una modificación en la estructura que contiene la información que él necesita no implica la modificación del programa.

¿Qué es una entidad?

En terminología de bases de datos se llama entidad al conjunto de ideas y hechos referentes a un ente concreto. La información sobre las entidades se divide en contexto, datos y representación de los datos.

¿Cuáles son las principales propiedades de un sistema de base de datos?

1. Consiguen la independencia entre datos y programas.
2. Facilitan la integridad de los datos, dado que todas las operaciones contra la base de datos se realizan a través de un gestor.
3. Confidencialidad de los datos. Igualmente, en ningún caso un programa podrá acceder a información para la cual no haya sido previamente autorizado en los bloques de control.



En la figura se representa un ejemplo típico de base de datos jerarquizada. Se trata, en este caso, de una base que contiene la información de un Centro de Estudios.

Estas son las principales instrucciones del lenguaje DL1, diseñado por IBM para la gestión de la base de datos jerarquizada IMS.

cilitar al usuario la instalación y el manejo de una base de datos jerárquica tanto en organización *batch* como en *on-line*.

Para la programación de los sistemas que actúan con la base es necesario utilizar un lenguaje anfitrión de alto nivel (COBOL, PL1, etc.) y en distintos puntos del programa se cederá el control de la ejecución a otros tipos de lenguajes específicos de la base de datos. Una vez ejecutadas las operaciones de consulta o modificación, el control de la ejecución volverá al programa principal redactado en un lenguaje de programación tradicional.

Como ya hemos señalado, en el entorno de las bases de datos jerarquizadas, al conjunto de campos de un determinado tamaño que contienen información de la base se le llama segmento. Y se llama registro a un conjunto de segmentos de tamaño fijo y jerarquizados, de forma que la relación entre los distintos segmentos de un registro será de «padre» a «hijo» o viceversa. Por último, se puede definir como base de datos jerárquica a un conjunto ordenado de registros físicos que, a su vez, están formados, como ya decíamos antes, por segmentos homogéneos clasificados.

Para aclarar el sentido jerárquico de este tipo de bases de datos podemos distinguir:

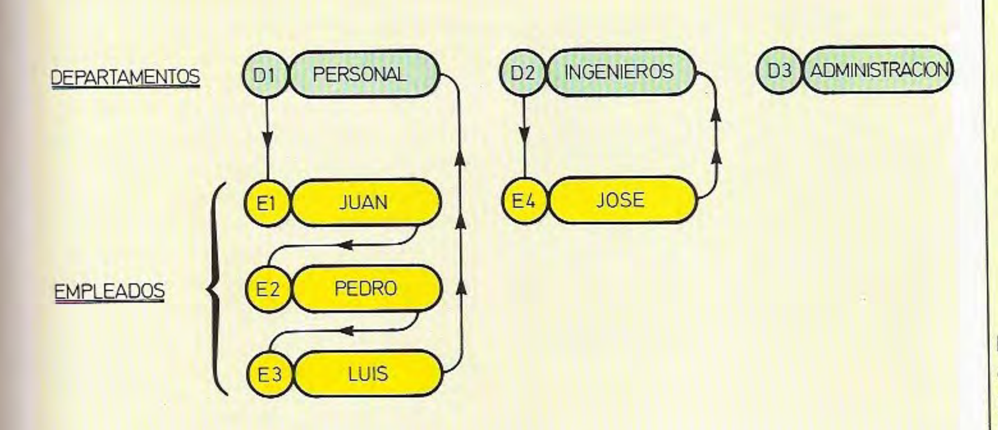
- Padre de un segmento. Un segmento A se dice que es padre de un segmento B si está estrictamente por encima de él.
- Hijos de un segmento. Se llaman hijos B_1, B_2, ..., B_N de un segmento A a todos los segmentos que están estrictamente por debajo de él.
- Segmentos hermanos. Se llaman segmentos hermanos a todos aquellos que son hijos de un mismo padre.
- Segmentos mellizos. Se dice que los segmentos B_1, B_2, ..., B_N son mellizos si son hijos del mismo padre y además son del mismo tipo.

El principal lenguaje de gestión para bases de datos IMS es el DL1, que está formado por una serie de procedimientos que pueden ser utilizados desde el lenguaje anfitrión para acceder a la parte del registro que interese en cada momento.

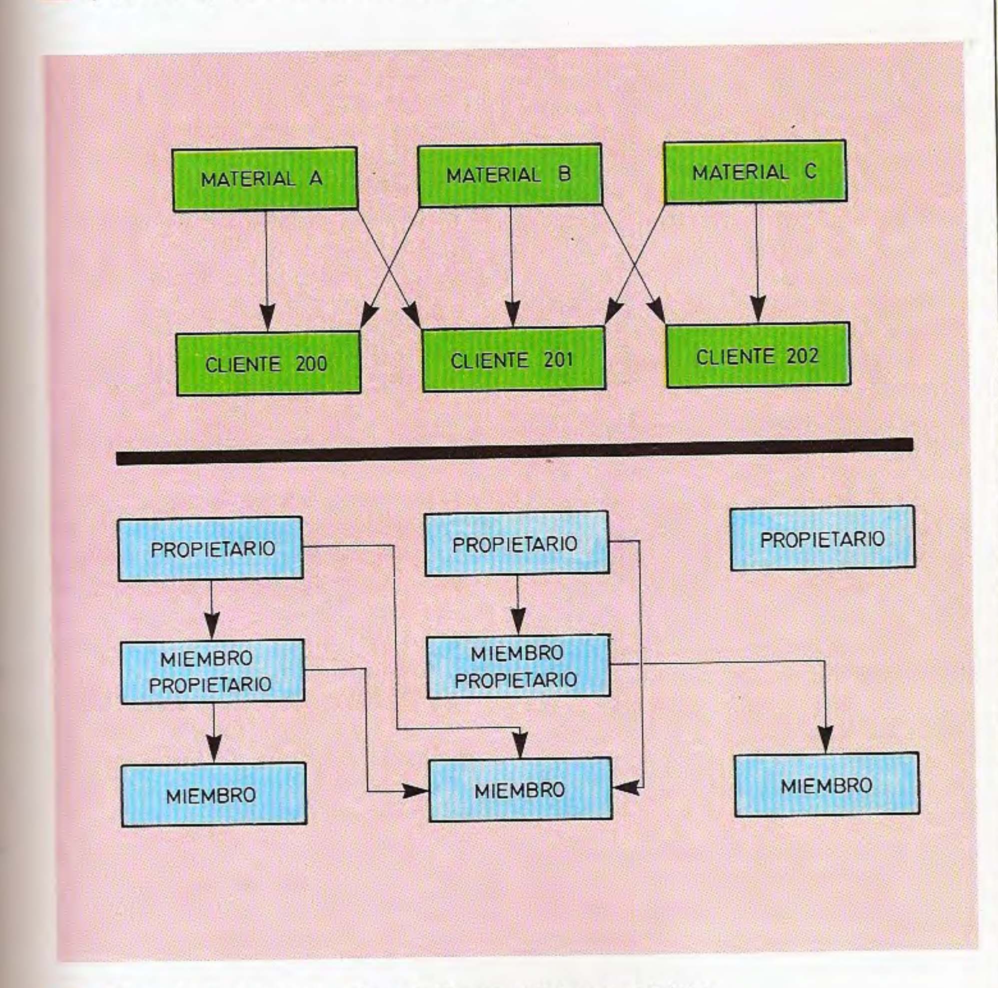
Modelo de red

El elemento básico de los modelos de red es el segmento, y la agrupación de los segmentos se hace mediante conjuntos, de forma que un miembro de un conjunto no puede pertenecer a ningún otro conjunto.

La jerarquía entre los distintos miembros de un mismo conjunto se indica



La figura corresponde a una base de datos de tipo red que contiene información de los empleados y de los departamentos de una empresa.



Ejemplo de estructura de red. El material A se suministra a los clientes 200 y 201; el material B a los 200, 201 y 202 y el C a los 201 y 202.

Administrador de la base de datos

La complejidad que lleva consigo la organización de un banco de datos hace que exista una persona, llamada *administrador de la base de datos,* cuya misión es organizar la base de datos, mantenerla, documentarla, cuidar de que se respeten las normas de seguridad y confidencialidad en el acceso a los datos, establecer los métodos de recuperación para el acaso de que haya una posible destrucción de los datos y reorganizar el banco de datos siempre que sea necesario.

Para llevar a cabo esta misión, que es evidentemente muy compleja, el administrador se ayuda de los programas y rutinas que existen en el banco de datos y de un lenguaje específico.

El administrador de la base de datos suministra a los usuarios, o programadores de la misma, un «modelo lógico» o subesquema por el que les limita el acceso a determinados datos.

Con este sistema el administrador logra también proteger la posible confidencialidad de los datos del banco, ya que es él quien, por medio del modelo lógico, controla los datos accesibles por el programador o el usuario.

El administrador de la base de datos es la única persona que conoce los tipos de organizaciones y de accesos de ficheros que tiene el banco de datos. Los usuarios sacan la información que desean del banco de datos, ayudándose del modelo lógico que les suministra el administrador. Es él quien fija también el modelo de estructura que debe tener la base de datos y que relaciona entre sí a los diferentes tipos de datos.

El administrador del banco de datos debe, en definitiva, ser una persona que conozca perfectamente todos los mecanismos de la empresa, de absoluta confianza y con una gran preparación informática, ya que es realmente quien controla todos los departamentos o divisiones de la empresa, quien conoce todos los procedimientos de almacenaje electrónico de datos, los lenguajes utilizados por los usuarios y programadores, etc.

CLIENTES

NUMERO	NOMBRE	POBLACION
200	PEREZ	MADRID
201	ROBLES	SEVILLA
202	FEITO	OVIEDO
203	PLA	GERONA

ARTICULOS/ PRECIOS

Nº ARTICULO	PRECIO
27	1500
30	750
33	12.420
34	5000

CLIENTES/ARTICULO

Nº CLIENTE	Nº ARTICULO	CANTIDAD
200	30	10
200	33	15
201	30	12
202	30	40
202	34	5
202	27	45

En las bases de datos con estructura relacional se utilizan tablas en las que, por ejemplo, se desglosan las características de los clientes y de los pedidos por éstos formulados.

mediante unos enlaces que dan a la base un aspecto de red, del que recibe el nombre. Los enlaces no sólo sirven para marcar un orden, sino que también se pueden emplear para almacenar alguna información.

Si la naturaleza de la información es tal que no existe una jerarquización entre los miembros de un mismo conjunto, se utilizan unos conectores para establecer la red.

Cada elemento puede depender de más de un elemento anterior. Un ejemplo de este tipo de estructura es el suministro de diversos materiales a otros tantos clientes. El material A se suministra al cliente 200 y al 201; el B, a los clientes 200, 201 y 202, etc.

Este modelo requiere, para su colocación física en el medio del archivo, la utilización de *apuntadores.*

Los segmentos, tanto en el modelo jerárquico como en el de red, pueden contener informaciones varias. Por ejemplo, el segmento cliente puede contener la fecha en que se realizó el pedido, condiciones de pago, etc.

Modelo relacional

El concepto básico de este modelo organizativo es la relación.

Utiliza tablas para su representación, de manera análoga a la de un fichero tradicional. Cada tabla consta de varias filas y cada fila tiene varias columnas. En las figuras se ve cómo se van desglosando, en forma de tablas, los diferentes pedidos de material que hacen los clientes. Cada tabla es como un fichero; las filas son los registros, y las columnas, los campos.

Formalmente se puede adoptar la siguiente definición: Sean D_1, D_2, ..., D_n n conjuntos llamados dominios, que no tiene por qué ser necesariamente distintos; una relación R sobre ellos es un subconjunto del producto cartesiano $D_1 \times D_2 \times \ldots \times D_n$.

Intuitivamente, una relación se puede asociar a una matriz en la que cada columna (dominio) contiene datos numéricos o no numéricos, y en la que cada línea representa un elemento de la relación a la que se suele denominar *tupla.*

Una vez definido el término relación, podemos decir que una base de datos relacional B es un conjunto finito de relaciones variables con el tiempo, definidas sobre un conjunto finito de dominios.

En un sistema de gestión de base de datos relacional integrado es usual que los distintos usuarios deban utilizar distintos subconjuntos del universo total de datos.

Se llama *modelo* al universo total de datos, es decir, al conjunto completo de todas las relaciones almacenadas en la base de datos, y se llama *submodelo* al conjunto de relaciones que pueden ser accedidas por un usuario determinado.

Para potenciar el rendimiento de una base de datos relacional es fundamental que las distintas relaciones que la componen estén adecuadamente estructuradas. El procedimiento para decir qué datos están en una relación y qué datos en otra se basa en el concepto de dependencia funcional. Veamos un ejemplo: sean dos colecciones de atributos A y B de una relación R. Se dice que B es fundamentalmente dependiente de A y se escribe A → B, si y sólo si en cada instante el valor de A en R está asociado con un solo valor de B.

Por ejemplo, si suponemos que en la relación R existen los atributos: Documento Nacional de Identidad (DNI) y Nombre; es seguro que NOMBRE depende funcionalmente de DNI (DNI → NOMBRE), ya que conociendo un valor concreto del DNI inmediatamente, mediante su tupla, se puede conocer el valor del NOMBRE correspondiente. Evidentemente la clave o identificador de una tupla estará formada por aquellos atributos de la relación de los que dependan funcionalmente el resto de atributos.

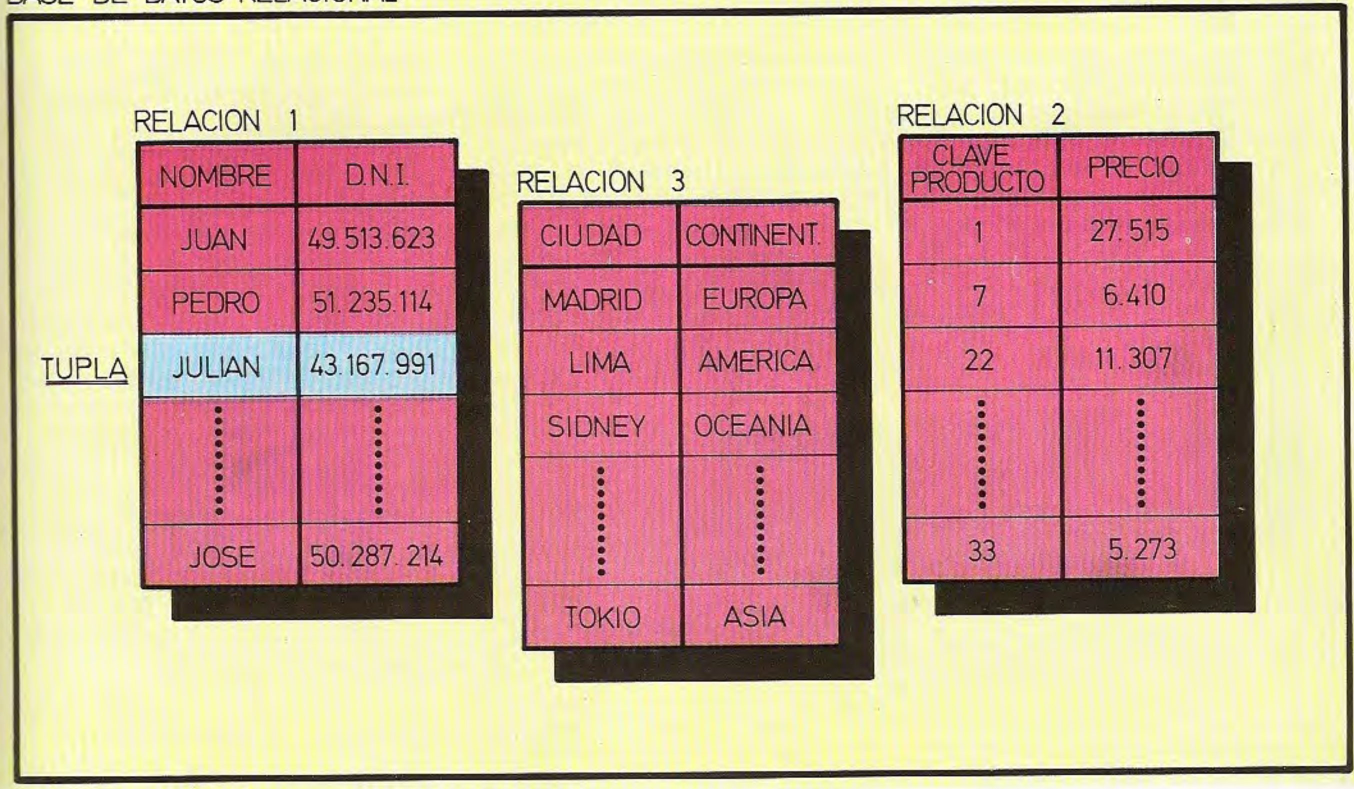
Para conseguir que las agrupaciones de datos en relaciones dentro de una base de datos relacionales sea óptima es necesario realizar una normalización. La normalización más extendida es la diseñada por Codd en 1970. Su principal virtud es que descubre las anomalías que puede presentar la base de datos en cuanto a actualizaciones, inserciones, eliminaciones...

El proceso de normalización se basa en una serie de formas normales (primera, segunda, tercera...) que proporcionan sucesivas mejoras en cuanto a la eliminación de anomalías.

La forma en que se gestionan las bases de datos es mediante uno de los diferentes lenguajes diseñados exclusivamente para esa misión. Entre los más importantes podemos citar:

1. *Lenguaje ALFA*

Basado en el cálculo de predicados, permite imponer una serie de restricciones que deben ser cumplidas por las tuplas de una o varias relaciones para aparecer en los resultados.



Las bases de datos relacionales están compuestas por distintas relaciones, las cuales a su vez, están formadas por tuplas elementales.

2. *Lenguaje QBE* (Query by example)

Se utiliza principalmente de forma interactiva y se basa en la exposición de un ejemplo del resultado esperado. A partir del ejemplo, el sistema comprobará las tuplas similares de la base de datos y procederá a representarlas como resultado.

GET W (VARIABLE): CONDICION	OBTIENE TODOS LOS ELEMENTOS "VARIABLE" QUE VERIFICAN LA CONDICION
UPDATE	ACTUALIZA UNA TUPLA YA EXISTENTE EN UNA RELACION
PUT	INCLUYE UNA NUEVA TUPLA EN UNA RELACION
DELETE	BORRA UNA TUPLA DE UNA RELACION

En la figura pueden observarse las principales instrucciones del lenguaje ALFA desarrollado para la gestión de bases de datos relacionales.

S

CLAVE	NOMBRE	CIUDAD	PROVINCIA
P. 3	P. LUIS	P. MADRID	P. MADRID

MEDIANTE ESTE EJEMPLO SE SOLICITA TODA LA INFORMACION REFERENTE A LAS TUPLAS DE LA RELACION S.

En el lenguaje QUERY BY EXAMPLE las informaciones se solicitan mediante un ejemplo.

Ventajas de una base de datos

Existen una serie de ventajas importantes al usar una base de datos en lugar de los sistemas convencionales de archivos. Algunas de ellas son:

- Consistencia de los datos. Derivada del hecho de que cada dato está una sola vez grabado en la base de datos. Si varios usuarios quieren conocer un dato, encontrarán todos el mismo valor.
- Un mantenimiento mucho más adecuado de los datos: cuando haya que actualizar un dato sólo debe hacerse una vez.
- Independencia de los programas de aplicación con respecto a la organización física de los datos y de sus métodos de acceso, ya que esta información sólo la conoce el administrador de la base de datos.
- Los costes del desarrollo y del mantenimiento son menores, ya que el trabajo del programador es más rápido y sencillo al no tener que ocuparse de los aspectos físicos del fichero, con el consiguiente ahorro de tiempo.

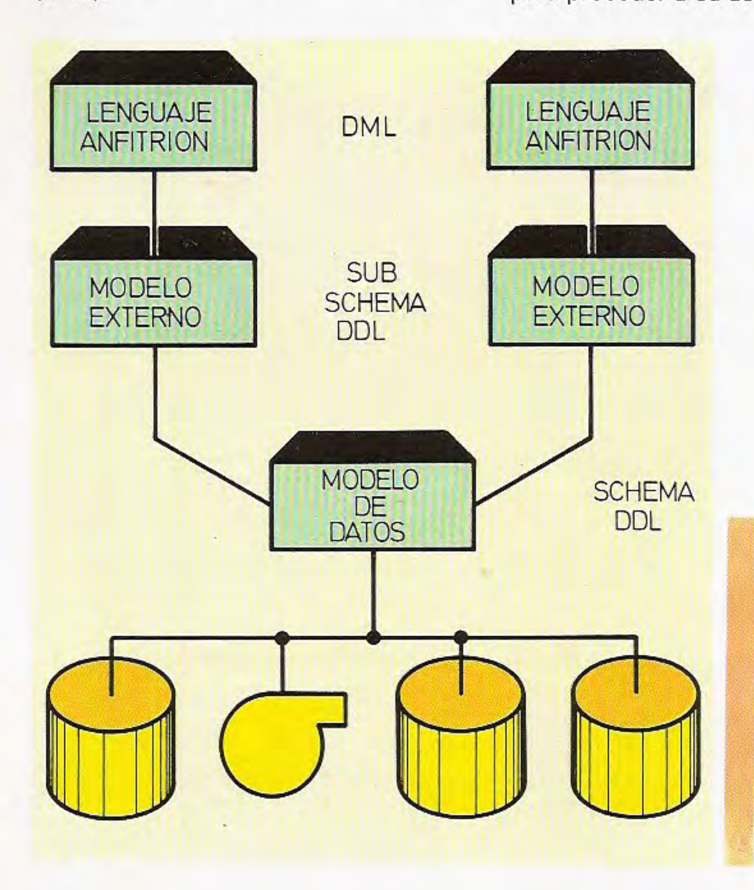
El grupo DNTG-CODASYL, después de un largo estudio de las mejoras que

se podían realizar sobre las bases de datos, llegó a la conclusión de que eran necesarios tres tipos de lenguajes:

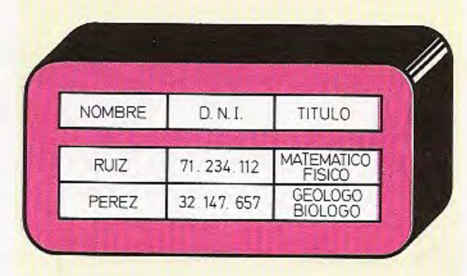
1. SCHEMA Data Definition Language (SCHEMA DDL).
2. SUB-SCHEMA Data Definition Language (SUB-SCHEMA DDL).
3. Data Manipulation Language (DML).

El primero de ellos sirve para realizar la definición de la auténtica base de datos, el segundo representa el modelo externo de la base de datos y el tercero es el lenguaje de consulta.

Conviene señalar que el lenguaje de consulta DML sirve tanto para extraer información de la base de datos como para proceder a su actualización.



Los tres tipos de lenguajes (SCHEMA DDL, SUB-SCHEMA DDL y DML) definidos por el grupo DBTG-CODASYL para la gestión de las bases de datos dan una visión de lo tres niveles lógicos existentes.



NOMBRE	D.N.I.	TITULO
RUIZ	71.234.112	MATEMATICO FISICO
PEREZ	32.147.657	GEOLOGO BIOLOGO

La figura ofrece un ejemplo de una relación que no está en primera forma normal.

NOMBRE	D.N.I.	TITULO
RUIZ	71.234.112	MATEMATICO
RUIZ	71.234.112	FISICO
PEREZ	32.147.657	GEOLOGO
PEREZ	32.147.657	BIOLOGO

La relación de la figura anterior se transforma en una relación en primera forma normal realizando los cambios que pueden observarse en esta figura.

Modelo de normalización para bases de datos relacionales

La teoría que vamos a describir fue diseñada por Codd en 1970 y persigue reducir las anomalías en la gestión de una base de datos relacional.

1. Base no normalizada

Inicialmente se parte de un diseño de relaciones que no cumple ninguna condición de antemano. A la base formada por estas relaciones se la denomina base no normalizada.

2. Primera forma normal

Se dice que una base de datos relacional está en primera forma normal cuando todas sus relaciones cumplen la propiedad de que cada tupla no contiene elementos que sean conjuntos. Es decir, cada dato de la relación es elemental.

3. Segunda forma normal

Se dice que una base de datos relacional está en segunda forma normal cuando todas sus relaciones cumplen las siguientes propiedades:

1. Son primera forma normal.
2. Todo atributo de cada tupla depende fundamentalmente de cada clave posible. Es decir, cuando todos los datos de cada tupla se pueden identificar cómodamente.

4. Tercera forma normal

Decimos que una base de datos relacional está en tercera forma normal cuando todas sus relaciones verifican las siguientes propiedades:

1. Son segunda forma normal.
2. Todo atributo de cada tupla no es transitivamente dependiente de cada clave posible.

Con ello se elimina el riesgo de que, al actualizar un atributo, no se actualice los que dependen transitivamente de él.

5. Otras formas normales

Existen muchas otras formas normales para una base de datos relacional. No obstante, cuando se consigue llegar a la tercera forma ya se puede garantizar que la gestión de la base será sencilla y eficiente.

Bases de datos para microordenadores

En la senda del ordenador personal



En cualquier oficina o en nuestra propia casa siempre existe una cierta cantidad de información que es preciso utilizar para sacar algún tipo de conclusiones. Si el volumen de datos a manejar es lo suficientemente grande como para que su manipulación y tratamiento no resulte fácil, nos encontramos ante la necesidad de una forma de organización automatizada. ¿Y qué más propio para ello que una máquina llamada ordenador, cuyo nombre revela bien a las claras sus aptitudes?

El problema puede ser, por ejemplo, el mantenimiento de una serie de citas de textos, con su autor, el título del libro, los temas más importantes de la cita, etc. Para tener toda esta información perfectamente organizada podemos usar varios archivos de fichas, cada una de las cuales sirve para llegar al texto de la cita.

La reunión de varios ficheros que comparten ciertos temas en común, o que se pueden relacionar para que sean utilizables en conjunción, recibe el nombre de base de datos.

La posibilidad de conformar una base de datos nos permite explotar en conjunto todos los datos que la forman. Las relaciones entre ellos son mucho más flexibles que en los ficheros convencionales, ya que están relacionados de forma lógica para el controlador de la propia base de datos.

En el momento de la creación de una base de datos se debe definir la estructura de las relaciones entre los ficheros que la componen.

Existen varias arquitecturas posibles de estas bases de datos, cada una con sus ventajas e inconvenientes. Básicamente se trata de sistemas que permiten llegar a los archivos de forma que el programa que accede a la base de datos no se tenga que preocupar de cómo está organizado el fichero ni de los campos que lo componen. Al programa le basta con saber que existen unos datos que se llaman de una forma determinada y que los puede utilizar.

Cualquier cambio que se efectúe en los archivos no implicará modificación alguna de los programas. Así pues, los datos y sus descripciones implícitas pueden asociarse a «puntos» simbólicos que los programas de gestión de la base de datos se encargan de controlar y entregar con la información correspondiente a los distintos programas externos que lo requieran.

Se puede relacionar este proceso con un interface lógico o unas cajas funcionales, de modo que una función de una de estas cajas sería la de reconocimiento y asimilación de las estructuras de la arquitectura de la base de datos, la cual se definió en el momento de la creación de la base a través de un lenguaje al efecto (por ejemplo, el DBMLS: Data Base Management Language System).

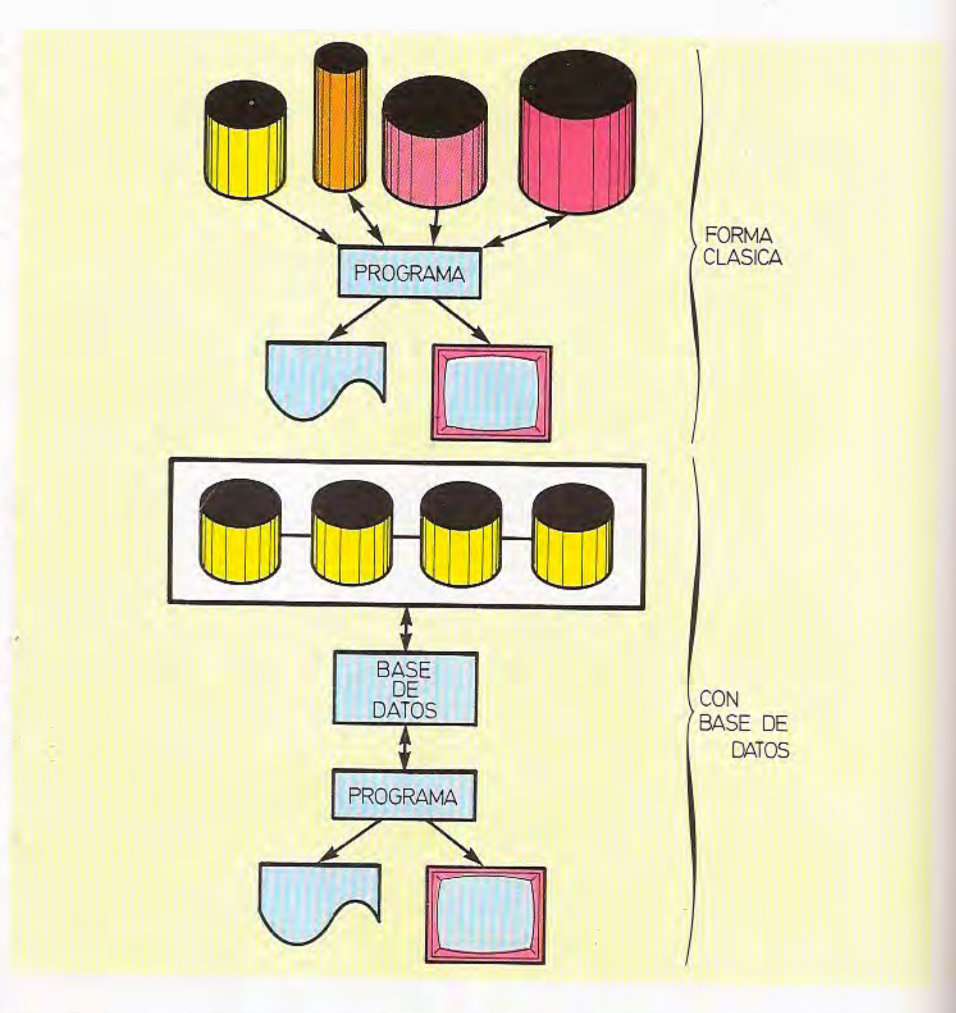
Después del reconocimiento del sistema de la estructura de los ficheros, otra «caja» se encarga de obtener la información que corresponda, según la «forma» que tenga la base de datos. También se encarga de la traducción de los datos simbólicos en toda acción referente a la BD. Por otra parte, el programa del usuario se encarga de dar los parámetros de entrada y tomar la información proporcionada.



Cuando en una oficina, o incluso en el hogar, se quiere manejar sin problemas una gran cantidad de información, lo normal es recurrir a un ordenador.

Otra forma de acceder a una BD es a través de un lenguaje de interrogación (QUERY), que permite explotar de una forma más flexible los datos existentes.

En resumen, un sistema de base de datos normal consta de una serie de programas para su creación, acceso, control y mantenimiento, además, claro está, del propio banco de datos.



En los sistemas de podemos llamar clásicos para el tratamiento de la información, el programa y los ficheros están íntimamente relacionados, de modo que una modificación en uno de ellos afecta al otro. Por el contrario, en una base de datos el programa que solicita una determinada información trata a los ficheros como si fueran todos iguales y sin que le afecte las modificaciones que en ellos tengan lugar.

El caso del microordenador

Básicamente, en los programas de bases de datos para microordenadores se introducen datos de una forma definida por el propio usuario sin mayores conocimientos de informática.

Esta forma de introducción de los datos es generalmente a través de una pantalla que formatea el propio usuario.

Existen otras pantallas que ya son del propio programa, que sirven para definir las acciones a efectuar con los datos residentes en los ficheros. Dichas acciones pueden concretarse en el establecimiento de relaciones de tipo aritmético y/o lógico entre datos almacenados. Podemos hacer que en pantalla aparezca un campo de suma de otros tres que no se ven, o sólo mostrar aquellos registros que tengan una fecha determinada.

La salida al mundo exterior de los datos puede ser a través de la propia pantalla o por impresora. En el primer caso el formato de salida quizá coincida con el de entrada de los datos, aunque es mucho más flexible y potente definir las pantallas de salida independientemente de la de entrada.

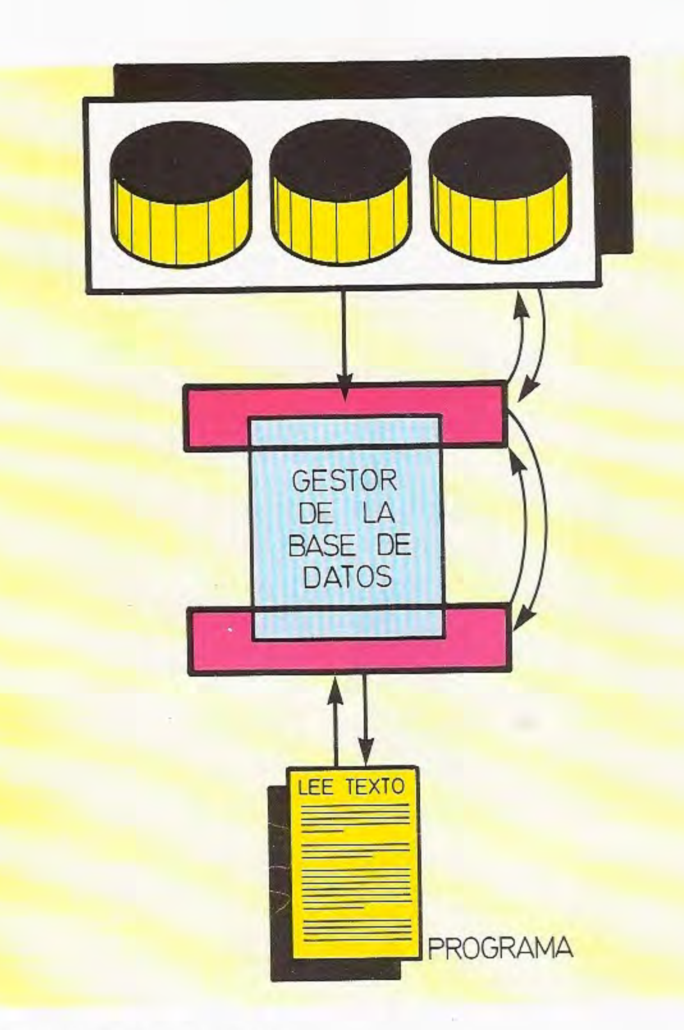
Por otra parte, son totalmente programables los campos que intervienen en el listado. Y, cosa muy importante, es posible mostrar por impresora sólo ciertos registros; es decir, el programa de base de datos debe poder seleccionar entre todos los registros que existan aquellos que cumplan ciertas condiciones.

Además de las tres funciones básicas: entrada, tratamiento y salida de la información, existen dos tareas bastante sofisticadas, pero muy útiles de cara a sacarle el mayor rendimiento a la información contenida en una base de datos. Se trata del diálogo entre ordenadores, con intercambio de contenidos y estructuras de las bases de datos propias, y la representación en modo gráfico de los contenidos o de los resultados de los procesos realizados en los archivos.

Una cuestión fundamental es que varias bases de datos puedan dialogar entre sí en la misma configuración, ya que, obviamente, la transportabilidad de la información es muy importante en esta categoría de sistemas.

Un último punto a considerar es que la relación con la base de datos debe hacerse de una forma lo más sencilla posible para el usuario. Para ello, el diseño de una pantalla debe poder realizarse a nivel de comandos o «dibujándola» directamente, dándole un nombre que permita posteriormente asociarla a una o varias bases de datos.

En cuanto a las órdenes que se quieran ejecutar, existen dos posibilidades: seleccionarlas a través de un menú o escribirlas directamente en la pantalla. Es importante en una base de datos la forma de rellenar una pantalla; la posibilidad de modificar cualquier campo con sólo situarse en él y no tener que especificar por medios complejos cuál de ellos, entre todos los posibles, queremos cambiar, así como la posibilidad por parte de la propia BD de comprobar tanto la validez del contenido de un campo que se acaba de introducir como si el



Los programas que solicitan una información acceden a la base de datos a través de un programa gestor que, actuando a modo de interface lógico, se encarga de la comunicación entre el programa y los archivos.

Para saber más

¿Cómo sabe el programa cuál es la capacidad de los ficheros de la base de datos?

Existen programas en los cuales es necesario indicar la cantidad de registros que va a tener el fichero, con el fin de reservar espacio en el disco o también para poner a los registros una marca de «libres» y luego poder usarlos normalmente. Si se desea ampliar la capacidad de los ficheros, suele haber utilidades dentro del propio programa que se encargan de ello.

¿Qué otras limitaciones pueden existir?

Una puede ser la cantidad de registros; por ejemplo, el programa dBASE II admite hasta 65.000 registros por fichero, siendo la máxima longitud del campo de 254 caracteres, mientras que la del registro es de 1.000 bytes. El programa Condor III tiene como máximo 32.000 registros, 1.024 caracteres de longitud de registro, 127 campos por registro y 127 caracteres de longitud de campo.

¿Es necesario ser profesional de la informática para explotar un sistema de este tipo?

La esencia de la informática no es saber programar, sino que el problema está en pensar informáticamente. Tener una visión informática de las soluciones y de la forma de traducirlas con las herramientas que se tienen... y esto sólo lo da la experiencia.

dato está dentro de un cierto intervalo, etc.

Algunas bases de datos para microordenadores

La verdadera popularización de las bases de datos ha sido paralela al desarrollo de la microinformática. Actualmente son innumerables los programas especializados en este cometido que se ofertan en el mercado.

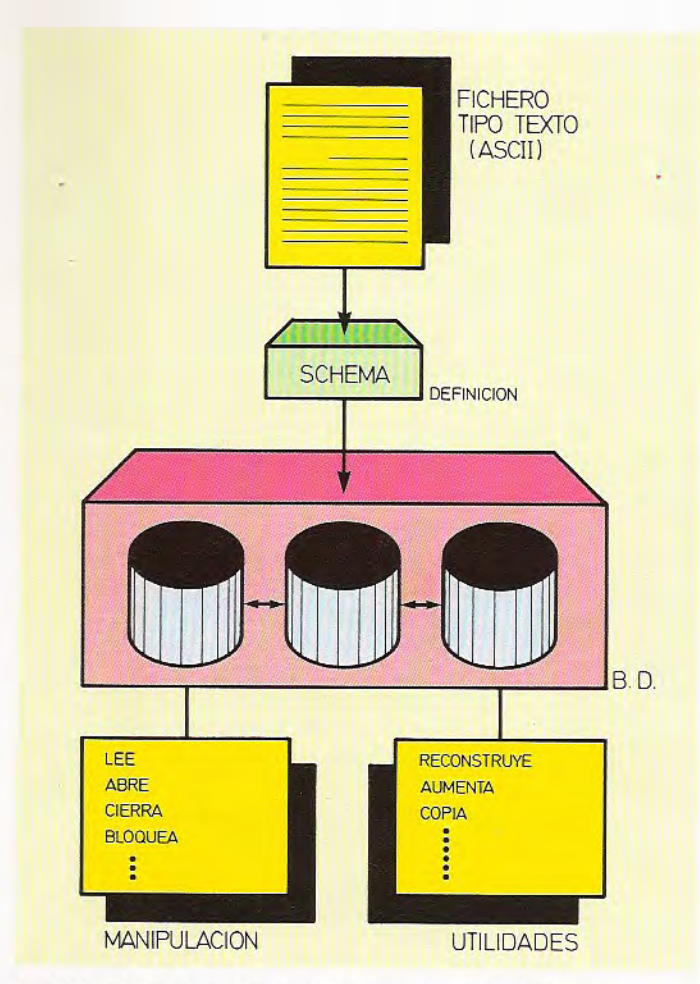
Sin lugar a dudas, el mayor protagonismo corresponde a la familia de «gestores de bases de datos» dBASE, de la firma estadounidense Ashton Tate. Su primer representante destacado fue dBASE II, programa que contaba con versiones tanto para ordenadores equipados con el sistema operativo CP/M como con MS/DOS. El relevo lo tomó dBASE III y, muy recientemente, la versión dBASE III Plus.

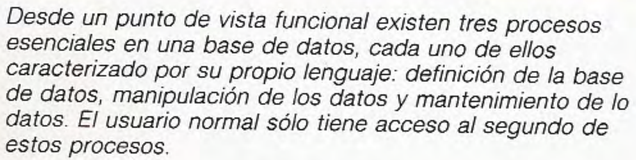
Otros programas para la gestión de bases de datos en microordenadores de resonancia en el mercado son R:BASE 5000, Revelation, PFS File & Report, DB-Master, Friday, Omnis, Infostar, Delta 4, Data Flex, DataStar...

La totalidad de los denominados paquetes integrados ofrecen asimismo un entorno de trabajo especialmente orientado a la gestión de bases de datos. Prueba destacable de ello lo constituyen los paquetes Lotus 1-2-3, Symphony, Framework, Jazz y Open Access.

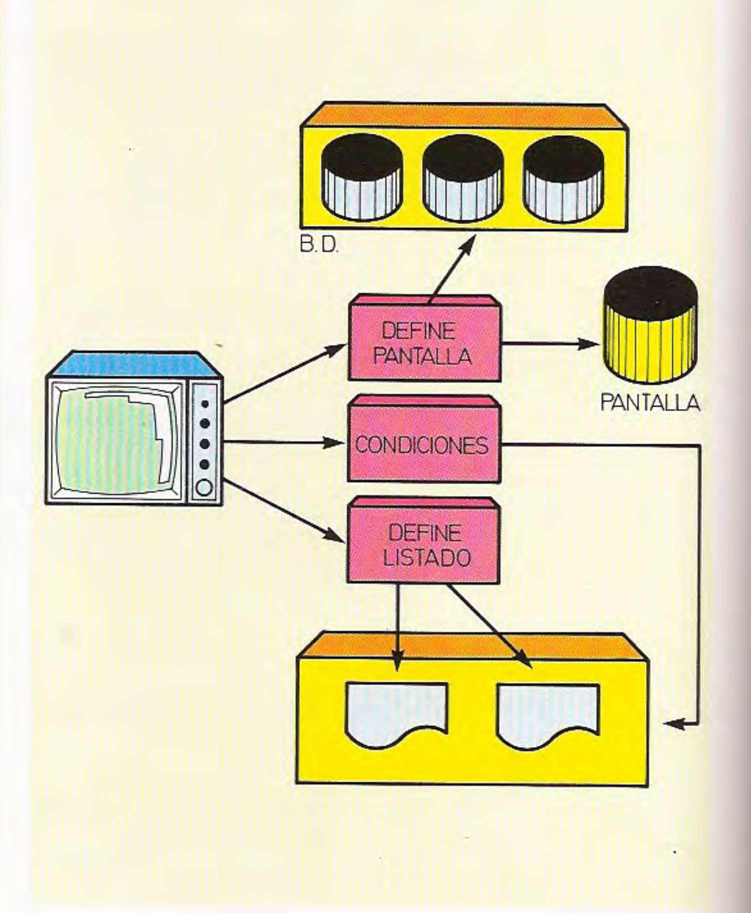
En estos casos, las bases de datos suelen compartir fácilmente información con otros entornos de trabajo, ya sea para facilitar la realización de cálculos, la representación gráfica de los datos almacenados o la edición automática de *mailings*.

La popularidad de los programas de esta índole ha alcanzado tal auge que incluso existen desarrollos —aunque muy limitados en potencia y capacidad de gestión— para microordenadores de tipo doméstico y educativo.

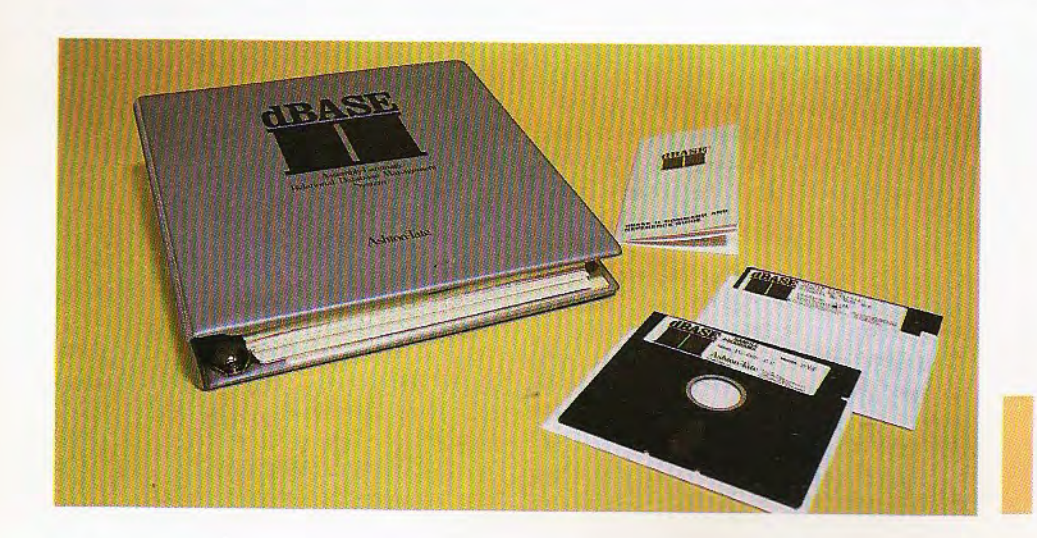




Desde un punto de vista funcional existen tres procesos esenciales en una base de datos, cada uno de ellos caracterizado por su propio lenguaje: definición de la base de datos, manipulación de los datos y mantenimiento de lo datos. El usuario normal sólo tiene acceso al segundo de estos procesos.



En un programa genérico de base de datos para microordenadores, las tres partes fundamentales son: definición de la pantalla, definición del listado y establecimiento de las condiciones para solicitar información.



La familia de programas dBASE II y dBASE III. Constituyen actualmente las soluciones más difundidas para la gestión de bases de datos con ordenadores personales.

Procesadores de texto

Gestionando la palabra escrita



a elaboración de texto es una de las actividades en las que más pronto se manifestó la eficacia de los sistemas informáticos. Su relevancia es tal que, hoy en día, no hay prácticamente ningún usuario de ordenador personal que no posea un programa especializado en el tratamiento de texto.

Un procesador de textos es un programa, habitualmente escrito en lenguaje de máquina para acrecentar su velocidad, que permite crear, corregir, formatear, guardar, recuperar y producir copias escritas de texto.

Pongamos como ejemplo el trabajo habitual en un despacho de abogados. En él se preparan informes escritos basándose en artículos de diversos códigos de justicia y en las resoluciones judiciales (jurisprudencia) sobre casos concretos. Toda esta información está organizada en voluminosos armarios de archivo repletos a rebosar.

Para empezar su trabajo, el abogado recupera la información oportuna del archivo y fotocopia las zonas de interés que servirán para elaborar su nuevo informe. Poco más o menos, su actividad se concretará en corregir o reelaborar documentos anteriores, añadiendo, eliminando o modificando palabras, frases o párrafos completos.

El siguiente paso es dar el *collage* confeccionado a una secretaria para que lo mecanografíe como documento final y lo incorpore al archivo interno para su uso posterior en otras tareas.

Como hemos visto, para realizar este trabajo con medios convencionales ha sido necesario contar con:

- Un archivo de información (o base de datos).
- Fotocopiadora u otro medio de reproducción similar.
- Tijeras, borrador, pegamento y otros útiles para modificar el texto.
- Una máquina de escribir para la puesta en limpio del documento y posterior vuelta al archivo.

La conclusión final no puede ser otra que constatar la enorme ventaja que supondría el trabajo con un procesador de



El proceso de textos está en primera línea de las necesidades de automatización en la empresa o en el entorno de actividad de los profesionales.

textos, el cual permite condensar el trabajo anteriormente descrito en un único proceso. Con dicha herramienta tendríamos en nuestras manos el medio de trabajo con más alto rendimiento laboral de la oficina.

Configuración del ordenador

Un ordenador dispuesto para el proceso de textos se compone básicamente de los siguientes dipositivos:

- *Pantalla*

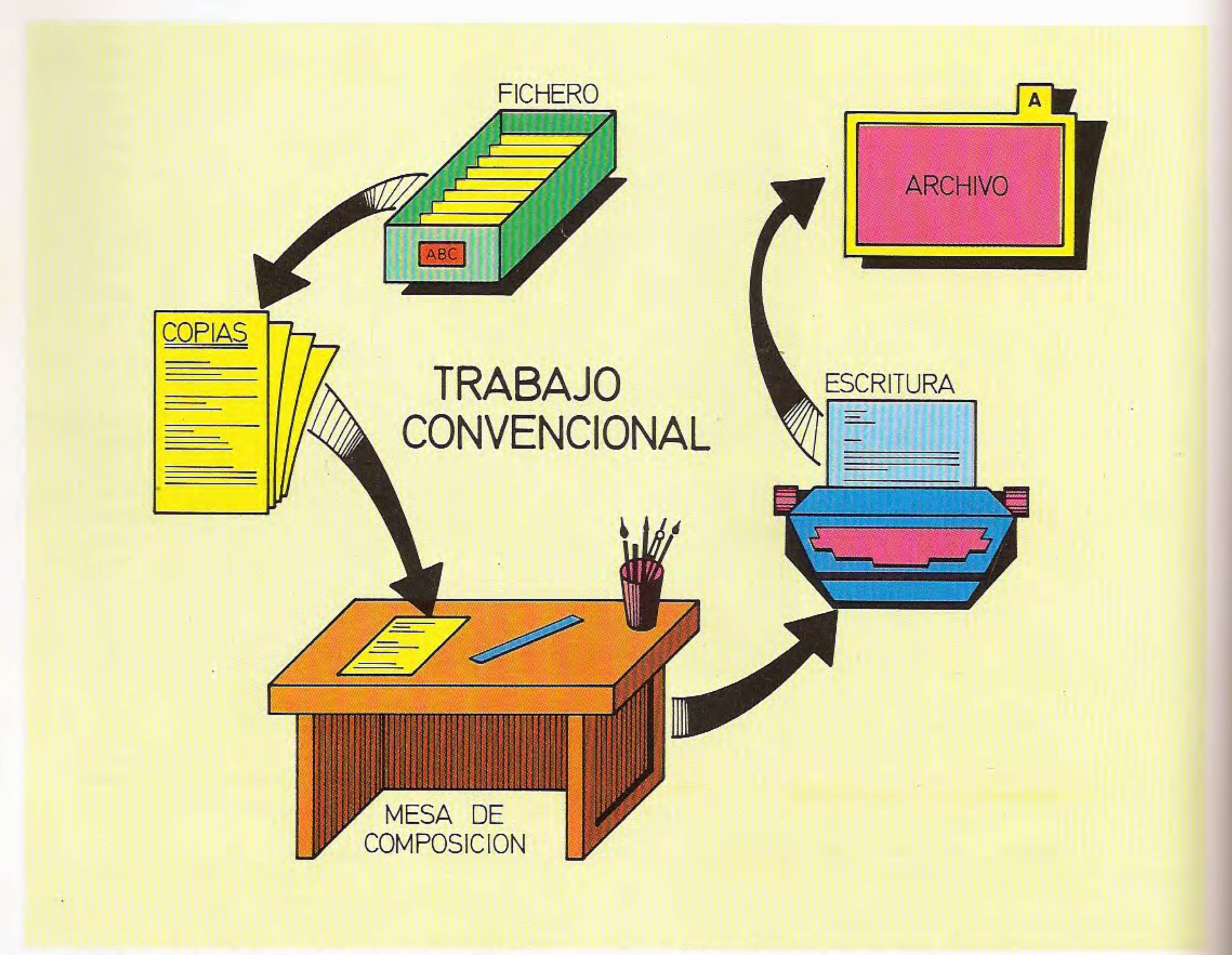
Habitualmente de 80 columnas y 25 líneas, aunque existen ordenadores equipados con pantallas de mayor capacidad de presentación: por ejemplo, 130 columnas y 60 líneas. Otros, de menor potencia, sustituyen la pantalla por un display apto para visualizar muy pocas líneas.

- *Unidad de disco*

Los medios de almacenamiento masivo más frecuentes son los discos y cintas magnéticas, discos flexibles o rígidos y cintas en casete en el caso de los ordenadores personales y domésticos. Los discos tienen la ventaja de admitir archivos más amplios, además de ser superior su fiabilidad y velocidad de acceso.

Los discos magnéticos normalmente utilizados con ordenadores personales para el proceso de texto podemos dividirlos en dos grandes grupos: los discos flexibles, con capacidades que van desde los 150 Kbytes hasta 1,5 Megabytes, dependiendo del tamaño y formato de los discos (que pueden ser de 3,5 o de 5,25 pulgadas), y los discos rígidos de tipo Winchester, que unen a su gran velocidad una más alta capacidad de almacenamiento.

La opción de archivo en casete, reservada a los ordenadores domésticos, tiene la ventaja de su bajísimo coste y el in-



El trabajo habitual en una oficina donde se maneja una gran cantidad de información, consiste en la elaboración de informes escritos, inicialmente en borrador, a partir de uno o vario ficheros; posteriormente los informes pasan a una fase de mecanografiado, previa a su entrada en el archivo para garantizar la reutilización del mismo.

conveniente de sus inferiores prestaciones en cuanto a velocidad y capacidad.

- *Impresora*

El tercer elemento de la configuración necesaria para el uso de un procesador de textos es el que permite la obtención de copias en papel. Las alternativas más frecuentes son las impresoras de matriz de puntos o de margarita, aunque no hay que olvidar el importante papel que están tomando las impresoras láser de sobremesa.

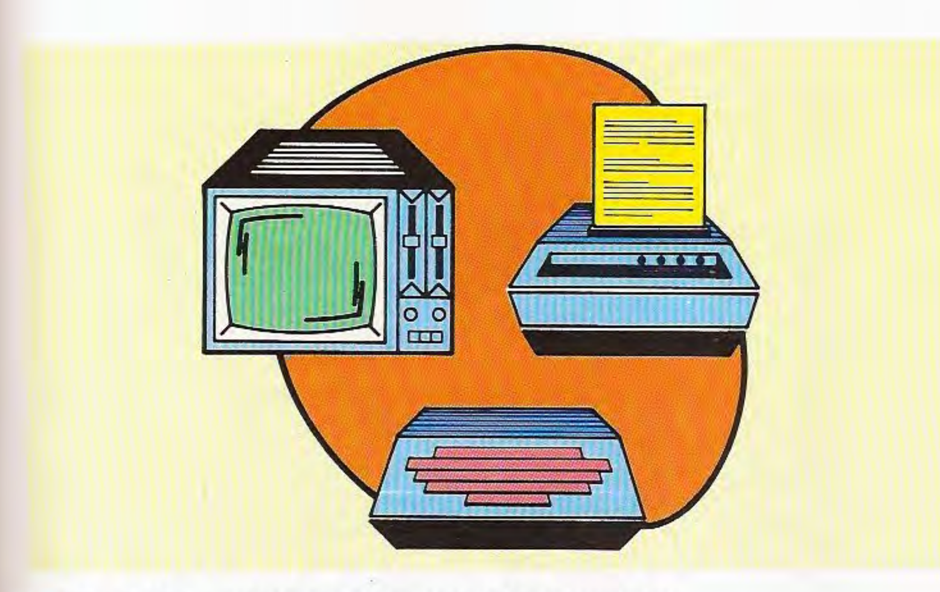
— Las impresoras de matriz de puntos tienen a su favor una más alta velocidad, coste inferior y capacidad de ofrecer varios estilos en la reproducción de los caracteres: cursiva, negrita, comprimida, realzada..., aunque su calidad no es equiparable a las de margarita o láser.

— Las impresoras de margarita ofrecen una calidad de letra semejante a la de una máquina de escribir electrónica, aunque sacrificando sus prestaciones en cuanto a velocidad y coste.

Un uso profesional del proceso de textos, por ejemplo, en el ámbito de la notaría tomada como ejemplo, puede exigir la adopción de una impresora de margarita. Por contra, si se trata de producir documentos de uso interno o documentos en los que la calidad no es un factor primordial, puede optarse por la alternativa más económica de la matriz de puntos.

Funciones

Las funciones básicas que ofrece un buen procesador de textos pueden dividirse en dos grandes grupos:



Mediante un procesador de textos la labor descrita en la figura anterior queda reducida a un único proceso. La simplificación es obvia.



En la actualidad existen equipos especialmente diseñados para el tratamiento de texto que combinan las funciones de una máquina de escribir electrónica con un software sofisticado.

— Funciones de escritura: Aportan las facilidades para escribir, formatear, modificar y borrar textos.

— Funciones de archivo: Permiten la grabación, recuperación e impresión de los documentos elaborados.

A continuación se describen algunas de las funciones de edición y archivo habituales en cualquier procesador de textos.

1. Funciones de escritura

Borrado: Permite borrar letras, palabras, líneas, párrafos o páginas completas.

Inserción: Debe permitir insertar caracteres, palabras, líneas y también párrafos. Es deseable la presencia de un comando de tipo MERGE que permita la inserción, dentro del documento con el que estamos trabajando, de párrafos o páginas de otros documentos archivados en el disco.

Repetición: Su presencia hará posible la repetición de párrafos dentro del texto elegido; igualmente, podremos mover líneas o párrafos completos de la posición en que se encuentran a la nueva que indique el usuario.

Edición: Incluye comandos que permiten:

— Crear cabeceras y pies de página repetitivos, de gran utilidad para libros o trabajos de gran volumen.

— Justificar los márgenes automáticamente, así como las tabulaciones definidas previamente (éstas pueden ser de dos tipos, alfabéticas y decimales).

— Definir la anchura del documento a tratar (número de caracteres por línea) y longitud de página (número de líneas que componen la misma).

— Generar tantas páginas como vayan ocupándose, con numeración automática de las mismas. Cualquier página que intercalemos producirá por sí sola una nueva numeración del conjunto.

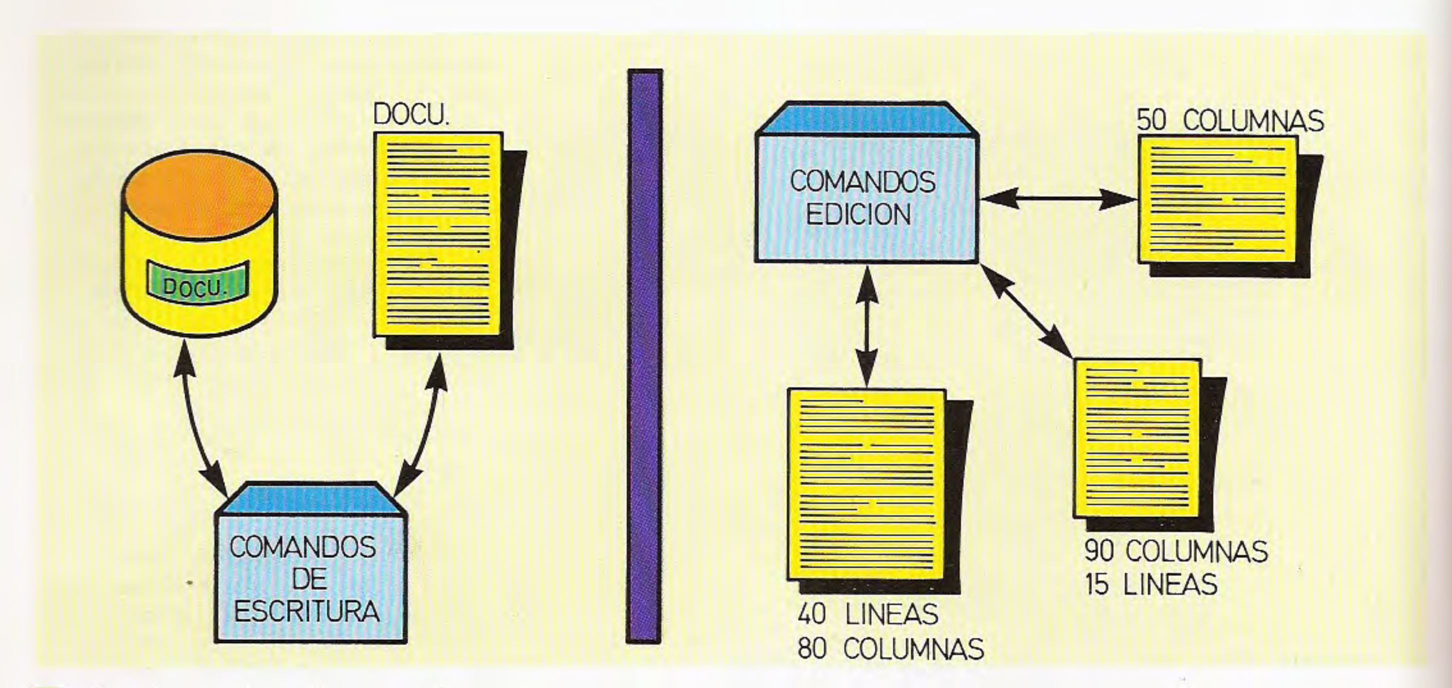
— Buscar un texto determinado y su posterior sustitución por otro que se defina.

— Obtener una relación de los ficheros previamente creados con el texto que tengamos en pantalla. Este comando es de gran utilidad en la edición de cartas o cualquier otro tipo de documentos que se desee personalizar, con ajuste automático de espacios.

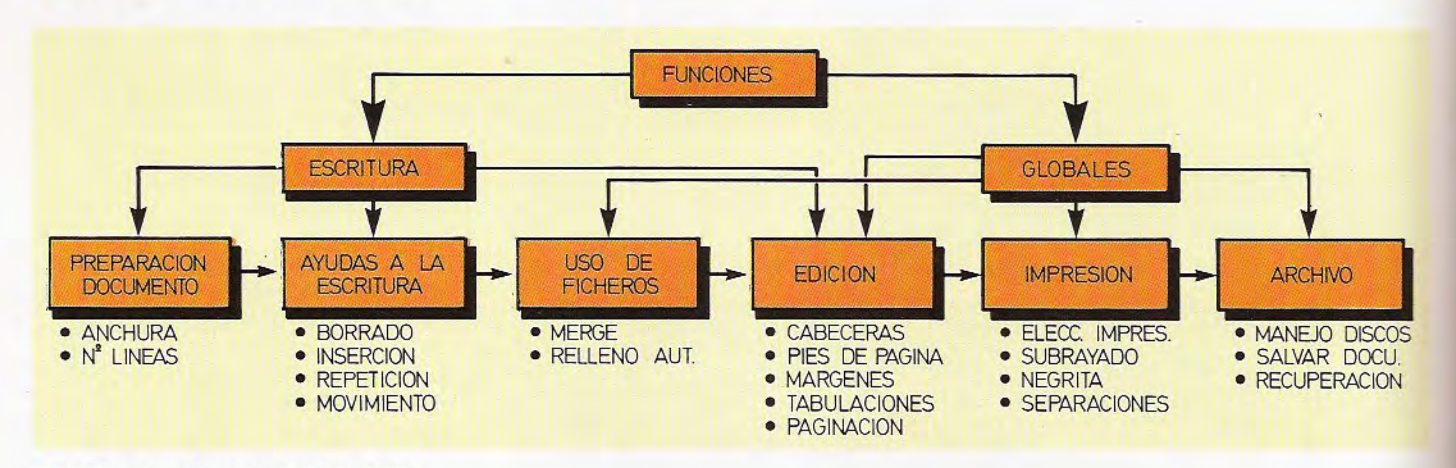
2. Funciones de archivo

Las funciones de archivo son las que facilitarán la relación del ordenador con los diversos periféricos conectados al mismo y la aplicación de órdenes que afecten al documento en curso globalmente. Algunos grupos de funciones catalogables en este apartado son los que siguen:

Archivo: Permiten almacenar en disco el documento editado para su posterior recuperación. En cualquier instante es posible archivar y recuperar los textos con el nombre que tengan asignado. Del



En todo tratamiento de textos existen básicamente dos tipos de comandos: los de edición y los de escritura.



Esquema de puesta en práctica de las diversas funciones de un procesador de textos.

mismo modo, podemos borrar documentos completos del disco basándonos en la instrucción oportuna.

Las funciones de archivo permiten igualmente formatear discos para su posterior uso, así como borrar discos completamente para su reutilización.

Impresión: Cualquier procesador de textos debe tener diversas opciones para seleccionar la impresora que se vaya a utilizar.

Debe admitir también la definición de parámetros tales como espaciado entre líneas y caracteres, impresión de caracteres con subrayado, sobreimpresión, en negrita, cursiva...

Las funciones de impresión deben permitir diferenciar entre si la impresión se va a efectuar sobre papel continuo u hoja a hoja. Esta opción debe poder alterarse según interese en cada momento.

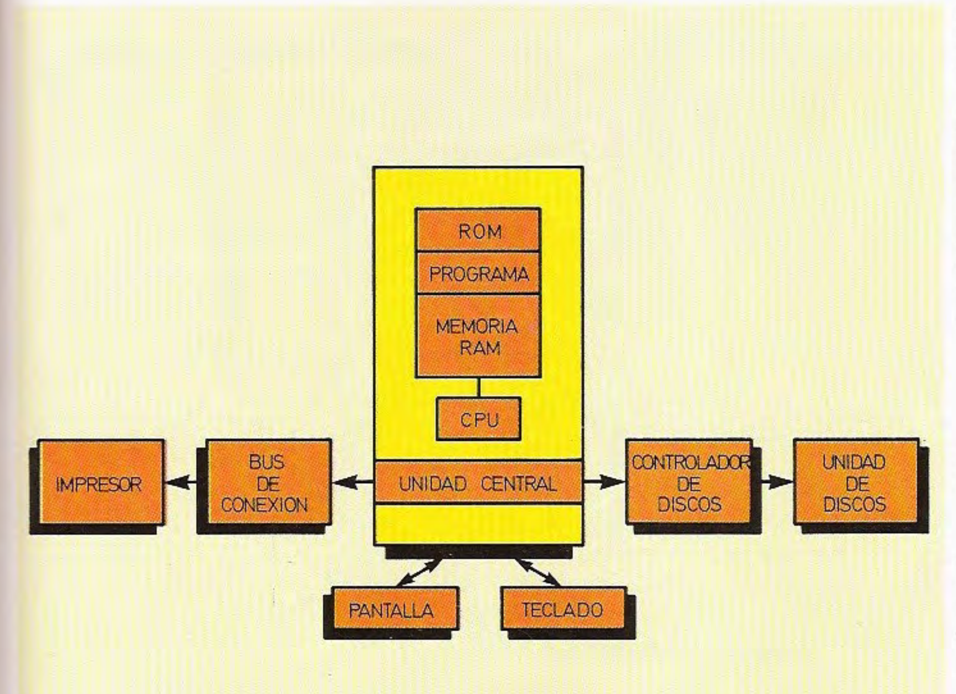
Instalación

Para el uso de un procesador de texto no es necesario ningún conocimiento especial, ni de lenguajes de programación ni sobre ningún otro aspecto relativo a la informática.

En orden a obtener el máximo provecho del procesador de texto es conveniente leer detalladamente los manuales de usuario que entrega el proveedor; es-

tos manuales están, normalmente, en castellano y pensados para personas sin conocimientos informáticos.

Hay algunas firmas que en el momento de la compra ofrecen cursillos de formación. Es muy recomendable acudir a los mismos si usted es un usuario recién llegado a este terreno. Con la asistencia a estos cursillos y con la ayuda del manual, el uso del procesador de textos puede ser rentable desde la primera semana de su instalación.



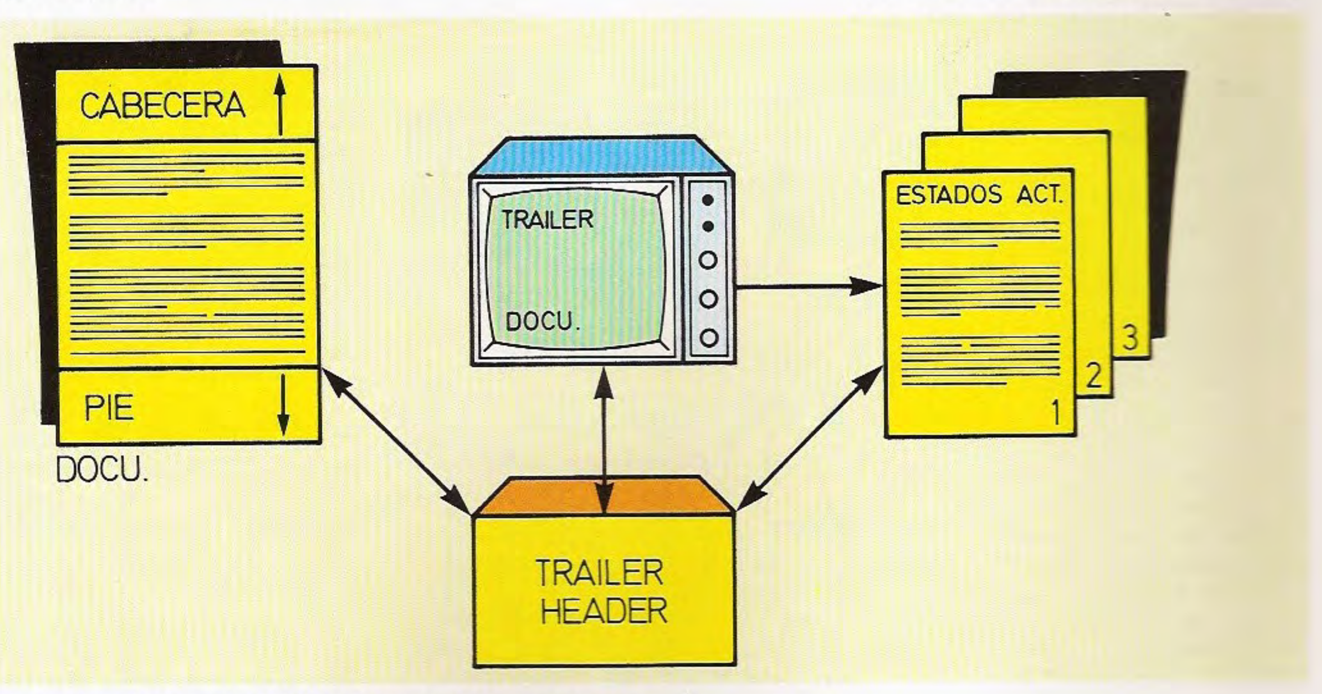
Esquema básico de las unidades que componen un sistema orientado al proceso de textos.

Conclusiones

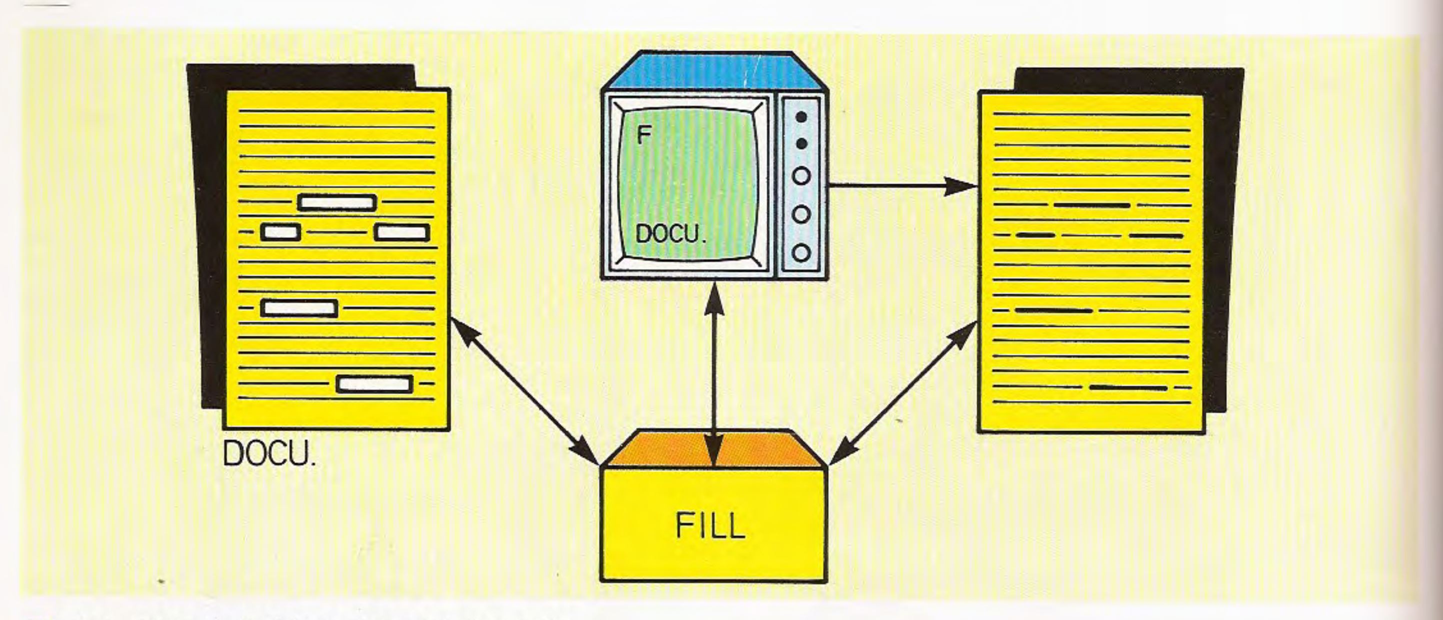
A continuación se resumen las diversas facilidades de trabajo que ofrece un procesador de textos, una aplicación especializada en apoyar la escritura, modificación, impresión, archivo y posterior recuperación de documentos:

- *Uso como máquina de escribir*

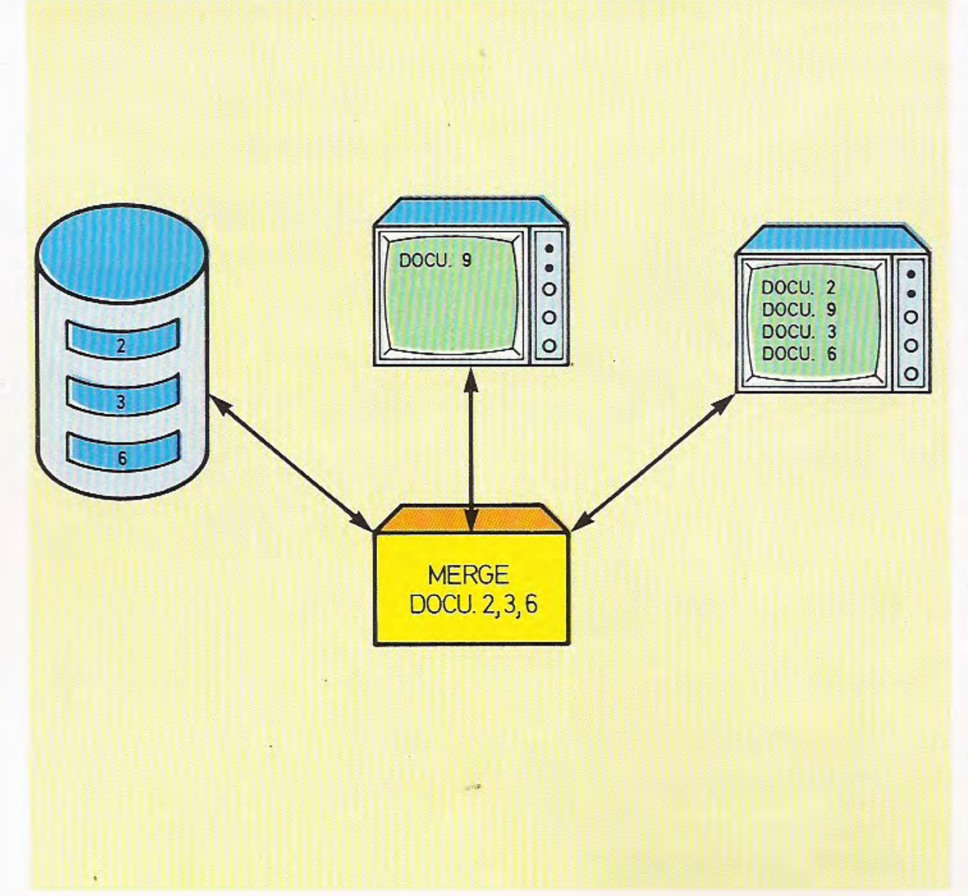
Permite una escritura rápida y cómoda. El texto se va visualizando en pantalla, facilitando la corrección instantánea de errores e incorporación de nuevos



En un documento de varias páginas la repetición de la cabecera y del pie de página a lo largo del mismo se consigue automáticamente apelando a los comandos al efecto.



Los puntos de anclaje permiten la introducción de texto en los huecos que se definieron como variables en la fase de edición.



textos, aparte de las facilidades para definición de márgenes, tabulaciones, etc.

- *Organización de textos*

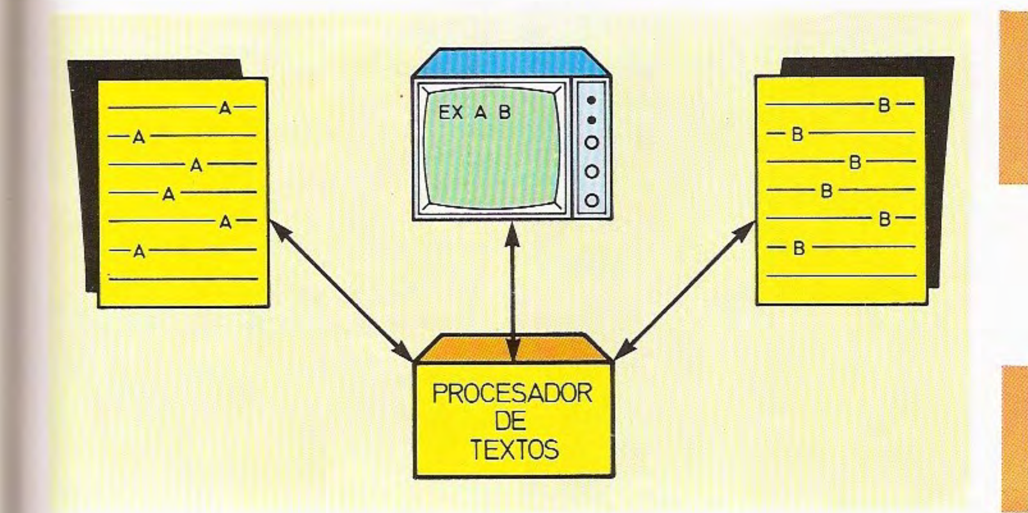
Con el uso del procesador no hay que ocuparse de la justificación de los márgenes a la derecha ni de palabras partidas con errores gramaticales. Contempla la paginación automática del texto de acuerdo con el número de líneas y columnas establecidas previamente.

Dispone, además, de numerosos recursos para enriquecer el documento: subrayado, cursiva, sobreimpresión...

- *Composición de textos*

Todas las facilidades anteriormente comentadas pueden solicitarse o desactivarse en cualquier momento de la escritura. Permite componer textos a partir de otros grabados anteriormente.

El comando MERGE permite crear un documento nuevo a partir de otros ya existentes en el disco. Los antiguos párrafos o documentos pueden introducirse en el orden y del modo que desee el usuario.



Algunos programas permiten la inserción, de forma rápida y automática, de un nombre, una dirección o cualquier texto que se vaya a repetir a lo largo del documento. Ello permite, por ejemplo, personalizar cartas.

En la figura se muestra en forma esquemática cómo se pueden utilizar las direcciones de un programa externo al tratamiento de textos para la elaboración de una circular predefinida.

FACTURACION

PROGRAMA
EXTERNO AL W.P.

DIRECCION

LUIS PONS

LUIS PONS

PROCESADOR
DE
TEXTOS



Hasta los más elementales microordenadores destinados al entorno doméstico o educativo cuentan con herramientas software para el proceso de textos.

- *Modificación de textos*

Además de la corrección por caracteres, existen instrucciones para borrar palabras, líneas o bloques cualesquiera de texto. Incorpora búsquedas de secuencias de caracteres, sustituyéndolas por otras distintas previamente definidas. Pueden reproducirse bloques de texto en cualquier posición.

- *Impresión de documentos*

Puede utilizarse una impresora de matriz de puntos, de margarita o láser, según el uso que se dé al procesador de textos. La impresión puede ser en papel continuo u hoja a hoja, eligiendo el tipo de interlineado y el cambio de tipo de letra, por el cambio de la margarita o la compresión o descompresión de la letra en las de matriz.

- *Archivo de documentos*

Se guardarán los diversos escritos con su nombre, organizados en páginas y capítulos, con la posibilidad de insertar bloques de textos, páginas o capítulos completos cuando se desee. De igual forma se puede recuperar toda la información almacenada en los discos.

- *Cartas*

Emisión de cartas personalizadas de forma automática.

- *Intercalado de párrafos*

Cuando aparecen en el texto bloques repetidos, éstos se pueden archivar como páginas de un documento normal y luego basta intercalarlos en los lugares del texto tantas veces como se quiera con una sola instrucción.

Existen algunos procesadores de texto que permiten al usuario visualizar el documento en pantalla en su formato definitivo, es decir, tal y como va a editarse el texto por la impresora. Esto ofrece la ventaja de trabajar en pantalla con subrayados, negritas o tipos distintos de letra.

Para saber más

Diferencias entre comandos de escritura y de archivo

Los comandos de escritura son los que se refieren exclusivamente a la elaboración del texto, siendo los segundos los que se ocupan de la relación entre la unidad central y sus periféricos. Un resumen válido sería que los de escritura permiten escribir, modificar y borrar textos, mientras que los de archivo facilitan la recuperación, impresión y posterior archivo de los textos.

¿Cuál es el cometido de la función MERGE?

Permite combinar líneas de programas. En los procesadores de textos este comando trabaja sobre la base de un texto en pantalla, al que permite incorporar otro documento previamente guardado en disco.

¿Puede interrelacionarse un fichero con un texto?

Esta es una de las aplicaciones típicas del tratamiento de los textos. Su uso en la edición de cartas personalizadas lo demuestra. Su forma de trabajo es la siguiente: se crea y archiva un fichero cualquiera (clientes, por ejemplo) y se empieza a elaborar una carta tipo, el uso de un comando nos permite la impresión de todas las cartas de forma totalmente personalizada.

¿Es útil la presentación en pantalla del formato definitivo?

La utilidad o no del mismo depende del uso que se dé al procesador de textos; si éste es para la edición de cartas, informes internos o cualquier otro tipo de trabajo interno, no parece ser de gran utilidad. Si el trabajo encomendado al mismo debe tener una edición perfecta, no cabe duda que es una opción altamente interesante, aunque esta ventaja por sí sola no debe influir en la elección del procesador.

Hojas electrónicas

La alternativa al lápiz, papel y calculadora



Las hojas electrónicas nacen ante la necesidad de adaptar los sistemas de planificación utilizados en los grandes ordenadores a los nuevos microordenadores.

Las primeras hojas electrónicas surgen a partir de una idea desarrollada en la Universidad norteamericana de Harvard. Su puesta en práctica corrió a cargo de dos programadores que, tras fundar una empresa de software, lanzaron al mercado el paquete denominado VisiCalc, la primera y más conocida hoja electrónica para microordenadores.

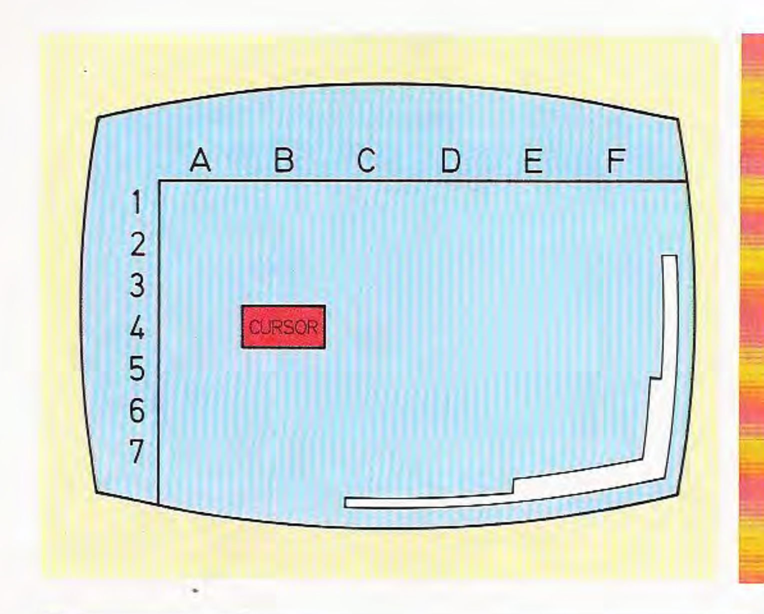
Podemos definir una hoja electrónica como una mezcla de calculadora, lápiz y papel, asociada a la gran capacidad de almacenamiento de datos y visualización de resultados que poseen los ordenadores personales.

Una hoja electrónica es comparable con una gran matriz de datos dividida en columnas y filas numeradas, y cuyas posiciones o celdas están definidas por sus coordenadas de fila y columna.

En cada una de estas posiciones puede introducirse un rótulo alfabético, una cifra o una fórmula de cálculo. La gran ventaja de esta herramienta de trabajo es que el ordenador recuerda las posiciones y las fórmulas previamente definidas. De este modo, si se cambia una cifra, todas las cifras posteriores relacionadas con ella se recalculan automáticamente.

Esta posibilidad de recalculación la convierte en una eficaz herramienta de trabajo para la elaboración de modelos financieros, planificaciones, control de ventas y/o costos, previsiones de tesorería, etc.

El uso de hojas electrónicas permite eliminar el papel, el lápiz y la calculadora a la hora de realizar estudios, planificar campañas o proyectar actividades de una empresa.



Una vez cargado el programa de una hoja electrónica aparecerá en pantalla una matriz igual o similar a la de la figura, donde las filas se designan con números y las columnas mediante letras. La posición del cursor queda referenciada mediante sus coordenadas (B4 en el caso de la figura).

	A	B	C	D
1	**DESCRIPCION**	**PRECIO**	**CANTIDAD**	**TOTAL**
2	Ordenador	9 NA	9 NA	[B3*C3]
3	Discos	9 NA	9 NA	[B4*C4]
4	Impresoras	9 NA	9 NA	[B5*C5]
5				
6			Total	[D3 + D4 + D5]
7				
8				

Campos alfabéticos: A-2 / A-3 / A-4 / A-5 / B-2 / C-2 / C-7 / D-2
Campos numéricos: B-3 / B-4 / B-5 / C-3 / C-4 / C-5
Campos fórmula: D-3 / D-4 / D-5 / D-7

En esta hoja electrónica se definen campos alfabéticos (de descripción), campos numéricos (de datos) y campos de fórmula (que indican las operaciones a realizar con los datos.

Una característica extensiva a la práctica totalidad de los programas de hoja electrónica es su sencillez de manejo; ello, unido a su robustez, permite al usuario no especializado en informática obtener el máximo provecho de esta aplicación.

Para trabajar con una hoja electrónica se debe empezar cargando el programa en el ordenador desde el soporte magnético en el que resida (normalmente, disco flexible o rígido). Tras ello, y después de entrar en la aplicación, aparecerá en pantalla una matriz ordenada por filas y columnas; el cursor aparecerá posicionado sobre uno de los elementos de la matriz (por ejemplo, en la celda A3: confluencia de la primera columna y tercera fila).

Nos podemos posicionar en cualquier coordenada de la matriz moviéndonos con las teclas de control del cursor. Y, una vez en la celda deseada, introducir títulos alfabéticos, números o fórmulas; los primeros no serán normalmente ejecutables, puesto que su función es meramente informativa de los datos que se van a introducir en la matriz. Los campos numéricos son los que aportan las constantes o datos para el funcionamiento de la hoja electrónica.

Por ejemplo, si se desea utilizar la hoja electrónica para elaborar una factura de venta de un determinado producto, éste podría ser el papel de los tipos de datos utilizables:

- *Campo alfabético:* Descripción del artículo.
- *Campos numéricos:* Precio unitario y cantidad.
- *Campo fórmula:* Total facturado.

Los campos de fórmula, una vez definidos, tomarán el valor que resulte de realizar las operaciones indicadas en la fórmula en cuestión, de acuerdo con los valores que tomen los campos numéricos que intervengan en la misma. La fórmula se recalculará cuantas veces sea necesario.

La posibilidad de recuperación de los datos es siempre por pantalla y/o impresora, siendo posible en ambos casos la presentación de los mismos en diversos formatos: ajustados a derecha o izquierda, en notación científica, con mayor o menor número de decimales...

Podemos, antes de su impresión, ver los formatos sobre la pantalla (uno de los problemas que tienen las hojas electrónicas es que la mayoría de las aplicaciones definidas sobrepasan visualmente el tamaño de la pantalla) y así comparar, por ejemplo, los datos de enero con los de diciembre.

A continuación se relacionan algunas ideas básicas sobre la utilidad práctica de las hojas electrónicas:

— Escandallos en procesos de fabricación, ofertas o seguimientos de costos.
— Planificación financiera con simulación de situaciones de mercado.
— Optimizaciones de resultados y/o producción.
— Estadísticas diversas.

Prácticamente todos los modelos de

	A	B	....	M
1	**INGRESOS**	**ENERO**		**DICIEMBRE**
2	Ordenador	10.000.000		19.500.000
3	Discos	12.500.000		23.750.000
4	Impresora	7.250.000		14.000.000
5	Total	29.750.000		57.250.000
6				
7	**GASTOS**			
8	Publicidad	5.000.000		12.000.000
9	Comisiones	2.425.000		6.500.000
10	Total	7.425.000		18.500.000
11	Ganancias	22.325.000		38.750.000

Debido a que el tamaño de algunas hojas electrónicas sobrepasa el formato de la pantalla, estas aplicaciones permiten comparar diversas zonas de una hoja mediante «ventanas».

Nombre hoja: ORDENADOR

	A	B	C	D	E	
1	DESCRIPCION	CANTIDAD	PRECIO ($)	PRECIO (PTAS.)	TOTAL (PTAS.)	
2	Microprocesador	1	15	2.400 (C2*160)	2.400	(B2*D2)
3	Circuitos	4	4	640 (C3*160)	2.560	(B3*D3)
4	Condensadores	8	—	400	3.200	(B4*D4)
5	Varios	3	—	150	450	(B5*D5)
6	Técnico	2	—	2.000	4.000	(B6*D6)
7	Soldador	6	—	1.500	9.000	(B7*D7)
8	Ajuste	1	—	2.000	2.000	(B8*D8)
9	Verificación ..	2	—	1.000	2.000	(B9*D9)
10				Total (ITE incluido)	26.762	[(E2+E3)*1,007+(E4+E5)*1,05+ +(E6+E7+E8+E9)*1,055]
11	PRECIOS PVP					
12	Precio 1	32.114 (E10*1,20)				
13	Precio 2	33.453 (E10*1,25)				
14	Precio 3	34.791 (E10*1,30)				

La figura representa la disposición de los diversos campos en una hoja electrónica del tipo VisiCalc. Se trata de obtener el precio de un ordenador a partir de los diversos componentes y mano de obra que intervienen en su fabricación.

ordenadores personales, e incluso los domésticos, cuentan con un amplio abanico de programas de hoja electrónica a su alcance, sea cual fuere su sistema operativo.

Algunos ordenadores personales, encuadrados en el segmento inferior, tienen una memoria central muy limitada y, por tanto, deben acceder permanentemente al disco en lugar de hacerlo a la memoria para evitar el problema de la capacidad. Ello presenta el inconveniente de una mayor ralentización tanto en el tiempo de elaboración como en el cálculo de las hojas de trabajo.

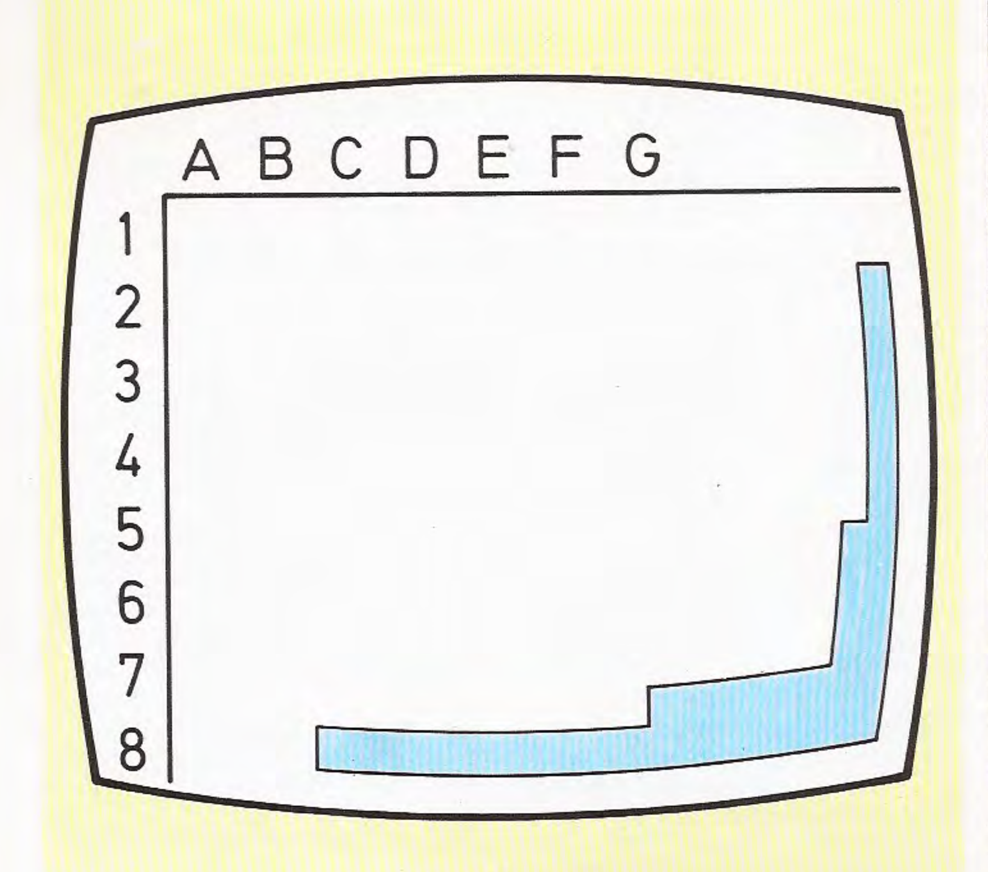
La configuración necesaria para el uso de una hoja electrónica suele reducirse a un ordenador personal cuya unidad central posea más de 64 kbytes, al menos una unidad de disco y complementado por una impresora para la presentación de los datos en papel.

VisiCalc, el precursor

VisiCalc es el nombre del primer programa de hoja electrónica que apareció en el mercado. Sus creadores fueron Bob Frankston y Dan Brickon, los cuales

Materia a elaborar	Ordenador.
Elementos que lo componen	Microprocesador de importación. Circuitos de importación. Condensadores (propios). Varios (propios).
Mano de obra	Técnico: dos horas. Soldador: seis horas. Ajuste: una hora. Verificación: dos horas.
Impuestos	ITE: 0,7 por 100, material ajeno. ITE: 5,0 por 100, fabricación propia. ITE: 5,5 por 100, servicios.
PVP	Primer precio: 20 por 100 de beneficios. Segundo precio: 25 por 100 de beneficios. Tercer precio: 30 por 100 de beneficios.

La hoja VisiCalc de la figura anterior se ha obtenido a partir de un cuadro de datos e hipótesis de valoración como el que se ofrece en esta figura.



Este es el aspecto de una pantalla con la matriz característica del programa VisiCalc.

Hojas electrónicas

Una hoja electrónica es una aplicación para ordenadores personales y domésticos destinada a usuarios no especializados en informática.
Se puede comparar básicamente a una matriz dividida y ordenada en filas y columnas, que nos permite interrelacionar unas con otras para la recalculación de datos.
La aplicación actúa sobre estas casillas, que pueden ser numéricas, alfanuméricas y fórmulas de relación, para simular supuestos contables, financieros, presupuestarios o cualquier otro que defina el usuario.
Las hojas electrónicas se explotan por medio de comandos y funciones predefinidas en el programa. Tanto unos como otras van elaborando un programa en código máquina para que el ordenador ejecute las instrucciones.
Las funciones más usuales son las siguientes:

- Logaritmos decimales y neperianos.
- Exponentes.
- Funciones trigonométricas.
- Máximos.
- Mínimos.
- Sumas.
- Valores financieros como LOOKUP.

Los comandos más frecuentes son:

- Borrado de caracteres, celdas o bloques; estos últimos bien por filas o columnas.
- Formatos de presentación, ajustes a derecha o izquierda, comas flotantes, notación científica, etc.
- Inserción de líneas y/o columnas.
- Carga y recuperación de formatos en disco con uso del comando MERGE.
- Comandos de ayuda como repetición de funciones o movimientos de una posición a otra de celdillas ya definidas.
- Posibilidad de fijar títulos o fórmulas para que no puedan ser modificados.
- Ventanas para solventar el problema de las hojas electrónicas grandes que no permite verlas completas en pantalla.
- Impresión de las hojas, en parte o completas, a definir por el usuario.

pusieron a punto la aplicación en San José (California), desarrollando una idea nacida en la Universidad de Harvard.

En los próximos párrafos se aportarán algunos detalles prácticos relativos a este programa de hoja electrónica, detalles que, aun superados hoy en día por programas mucho más potentes y evolucionados, servirán sin duda para conocer la filosofía de trabajo de esta categoría de aplicaciones.

Para utilizar debidamente la hoja VisiCalc el usuario debe estar, de antemano, perfectamente familiarizado con el teclado de su ordenador. Las teclas que se van a utilizar más habitualmente son las siguientes:

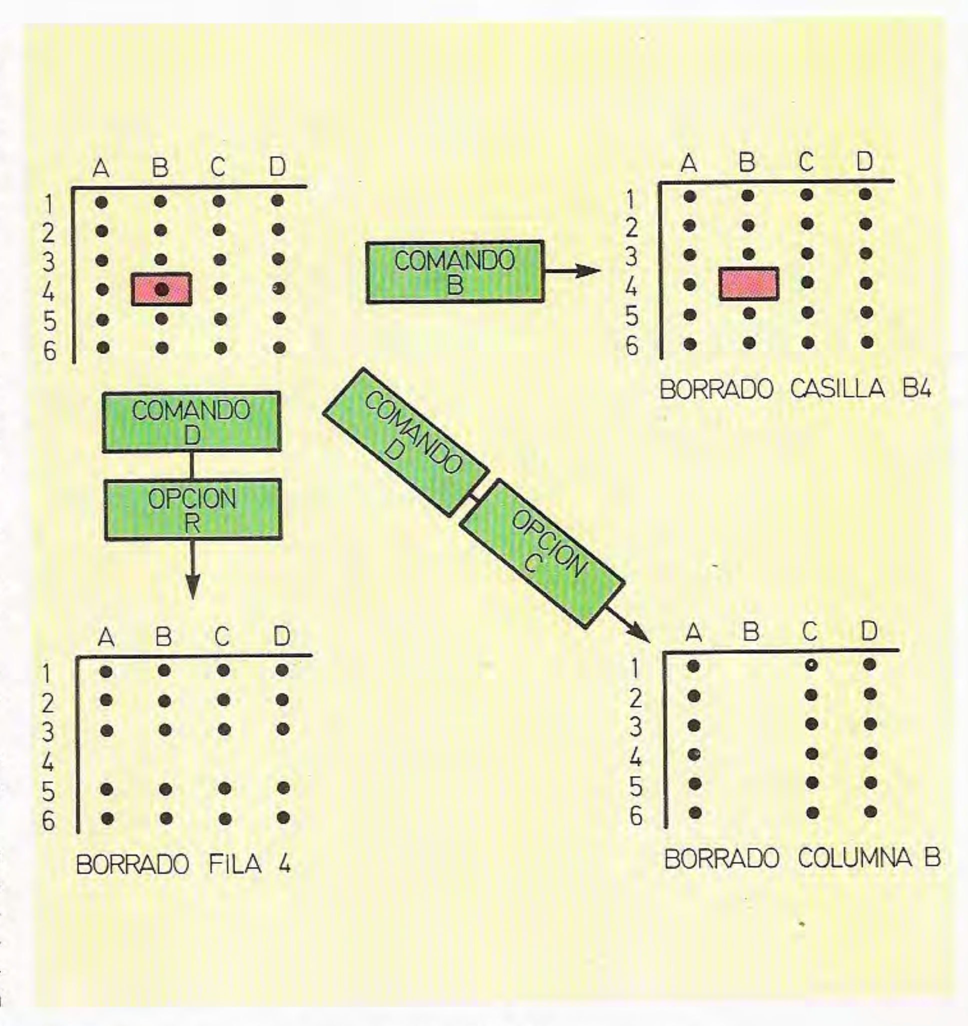
— De movimiento del cursor: arriba-abajo y derecha-izquierda.

— De coordenadas para movimiento rápido del cursor. En nuestro estudio es la tecla «>» (mayor que).

— La barra inclinada, «/», la cual presenta en la pantalla los diversos comandos que utiliza el VisiCalc.

— La tecla RETURN, cuya misión es poner fin a la entrada de cada dato o comando.

La sesión empieza al introducir en la correspondiente unidad lectora el disco flexible que almacena el programa VisiCalc. Una vez que se llame a la aplicación, aparecerá en la pantalla del ordenador un mensaje indicando que se está cargando el programa. Poco más tarde, la visualización mostrará la tradicional matriz de trabajo distribuida en filas y columnas.



La figura representa esquemáticamente el funcionamiento de los comandos de borrado de celda (comando B) y de fila y columna (comando D).

Comandos de VisiCalc

Todos los comandos de VisiCalc se ejecutan precedidos por el carácter «barra inclinada» (/). Una vez introducido dicho carácter, se puede elegir entre las siguientes opciones: B, C, D, F, G, I, M, P, R, S, T, W, que corresponden a los diversos comandos de la aplicación; éstos se describen seguidamente:

- B: Borrado de casilla. Borra la casilla sobre la que está situado el cursor.
- C: Borrado de hoja. Borra la hoja VisiCalc en curso residente en la memoria central, sin alterar las archivadas en disco. Este comando se ejecuta previa confirmación del mismo con la tecla Y.
- D: Borrado de líneas. Permite borrar la fila o columna en la que está situado el cursor. Una vez pulsado el comando D, el ordenador pregunta si se trata de fila o columna.
- F: Formato de casilla. Este comando se ejecuta sobre cada casilla y tiene diversas opciones. Una vez pulsado el comando F, el ordenador ofrecerá las siguientes posibilidades:

— D: Formato en coma flotante. Cada campo aparece con los decimales que le corresponden.

— I: Formato en números enteros. Redondea los decimales.

— L: Ajusta los números al margen derecho de la casilla.

— R: Ajusta al margen izquierdo.

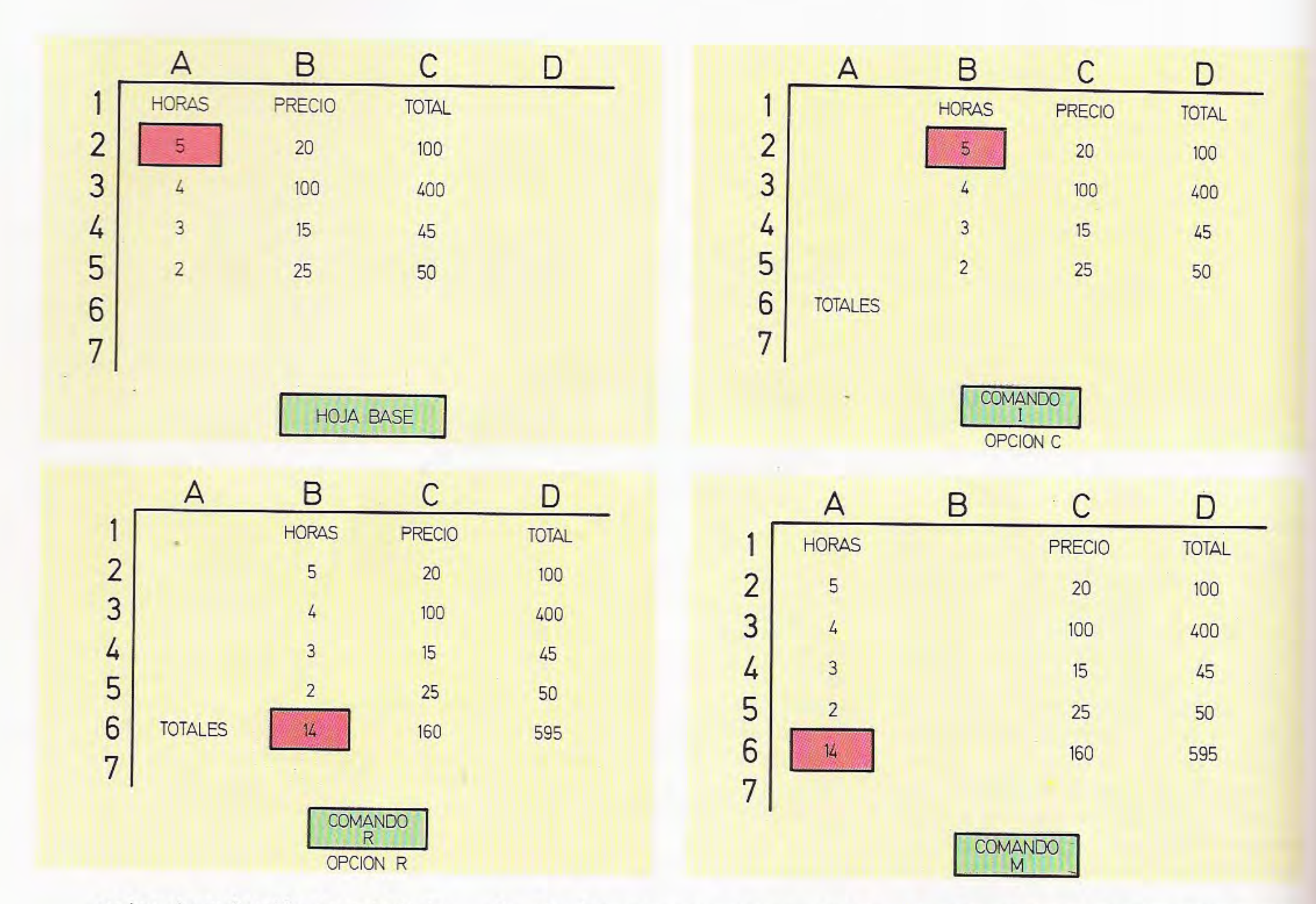
— $: Formato con dos decimales.

- G: Formato global. Esta opción permite preparar la hoja VisiCalc para la recepción de datos; a su vez, ofrece las siguientes posibilidades:

— C: Anchura de cada casilla, con un mínimo de tres caracteres.

— R: Recalculado de la hoja, que puede ser activado de forma manual o automática.

— F: Orden de formato global. Está subdividida en los mismos apartados que el comando F, sólo que su



Esquema de funcionamiento de los comandos I, M y R. A partir de una hoja base con el cursor en la posición A2 se quiere introducir en la fila sexta una línea de totales. Mediante el comando I se inserta una columna, lo cual permite introducir el literal TOTALES en la posición A6. Asimismo, se introduce la fórmula B2 + B3 + B4 + B5 en la posición B6, con la opción relativa para las posiciones C6 y D6, lo cual nos permite obtener los totales de las tres columnas existentes en pantalla. Mediante el comando M es posible mover una fila o una columna a una nueva posición.

orden de aplicación es a toda la hoja en lugar de a sólo ciertas casillas.

Un ejemplo del funcionamiento de estos comandos es el que se apunta a continuación: se trata de crear una hoja electrónica con una anchura de columna de 12 caracteres en la que trabaje con números reales (coma flotante), exceptuando los campos de totalización que deben aparecer en formato entero. Los literales deben aparecer ajustados a la izquierda.

Para cumplimentar este supuesto deberíamos pulsar los siguientes comandos:

1. G, de global; C, de anchura, y 12 para definir el tamaño.
2. G, de global; F, de formato; D, de coma flotante.
3. En los campos que tengan definido un total se deberá pulsar, casilla a casilla: F, de formato, e I, de números enteros.
4. En los campos alfanuméricos donde haya un literal se pulsará F, de formato, y R, de ajuste a margen izquierdo.

Una vez formateada la hoja VisiCalc, el usuario puede echar mano a una serie de comandos que resultan de gran ayuda para la elaboración de supuestos; éstos son los que se detallan a continuación:

- I: Inserción. Permite insertar filas o columnas respecto a la posición del cursor, respondiendo a la interrogación R o C.
- M: Mover. Mueve la fila o columna en la que se encuentra el cursor a la nueva posición que se le indique. Esta nueva situación se puede especificar tecleando las coordenadas de su nuevo destino o moviendo el cursor a través de la matriz.
- R: Repetición. Este comando es el que permite pasar información contenida dentro de una casilla o columna a otra casilla o columna de la propia hoja. Cuando el campo que se quiere duplicar es una fórmula, preguntará si cambian o no las coordenadas a que afecta esta fórmula en su nueva posición.
- T: Fija títulos. Permite crear unos espacios fijos a los que no se podrá ac-

ALGUNAS HOJAS ELECTRONICAS					
SISTEMA OPERATIVO	HOJA ELECTRONICA	MEMORIA	SISTEMA OPERATIVO	HOJA ELECTRONICA	MEMORIA
ATOM	ATOMCALC	12 K	MS-DOS	MULTIPLAN	56 K
				MICRO-FINAR	64 K
COMMODORE	BUSICALC	16 K		LOGISCALC	64 K
	CALCRESULT	23 K		PEACHCALC	44 K
	EASYCALC	64 K		PLAN-80	56 K
	PRACTICALC	16 K		SCRATCH-PAD	48 K
				SUPERCALC	64 K
CPM	CALCSTAR	56 K		SPREADSHEET	64 K
	IMPS	48 K		T/MAKER	48 K
	MULTIPLAN	56 K		UNICALC	64 K
	MASTERPLANER	64 K			
	MICRO-FINAR	64 K	OASIS	MULTICALC	DISCO
	PEACHCALC	44 K			
	PLAN-80	56 K	SHARP	EASYCALC	8 K
	PLANNERCALC	64 K			
	SCRATCH-PAD	48 K	VARIOS		
	SUPERCALC	64 K	SISTEMAS	VISICALC	48 K
	SPREADSHEET	64 K			
	T/MAKER	48 K			
	UNICALC	64 K			

ceder. Su utilidad es la de evitar errores en la entrada de datos. Los títulos se pueden fijar, bien horizontal o verticalmente, tecleando las letras H o V, respectivamente.

• W: Ventanas. Como quiera que las pantallas de los ordenadores tienen habitualmente una resolución de 25 líneas por 80 columnas y las hojas electrónicas rebasan estas dimensiones, nos encontramos con el problema de comparar los datos de un extremo de la hoja con los del otro. Al efecto se pueden definir en nuestro caso hasta dos ventanas, esto es: se pueden separar dos zonas de la hoja y mostrarlas en la misma pantalla.

Estas ventanas pueden definirse de forma horizontal, con H, o vertical, con V. A su vez, su paso puede sincronizarse, con la orden S, o desincronizarse, utilizando la letra U.

El último grupo de comandos VisiCalc que vamos a presentar es el que se refiere a las relaciones de la aplicación con los periféricos del ordenador, básicamente con la impresora y con la unidad de discos. Estos últimos son los que nos ocuparán en primera instancia:

• S: Discos. Está dividido en las siguientes opciones:

— L: Cargar. Esta posibilidad recupera del disco una hoja previamente grabada, con los datos y fórmulas asociados a cada casilla.

— S: Almacenar. Guarda en disco la hoja electrónica en curso. Debe almacenarse en el disco otorgando previamente un nombre a la hoja.

— D: Borrar. Borra el fichero de hoja electrónica especificando, previa confirmación del comando con la tecla Y.

— I: Inicializar. Prepara nuevos discos para trabajar con ellos, formateándolos para absorber información VisiCalc.

— Q: Salir. Finalización del trabajo; exige confirmación con la tecla Y.

— #: Ficheros. Esta es una de las opciones más importantes de VisiCalc, semejante a la orden MERGE del lenguaje BASIC. Permite almacenar o recuperar del disco una parte o toda una hoja VisiCalc para integrarla en otra de orden superior.

En el caso de operar con ficheros DIF se tiene la desventaja de que pasan los datos, pero no las fórmulas asociadas a los mismos.



La hoja electrónica VisiCalc

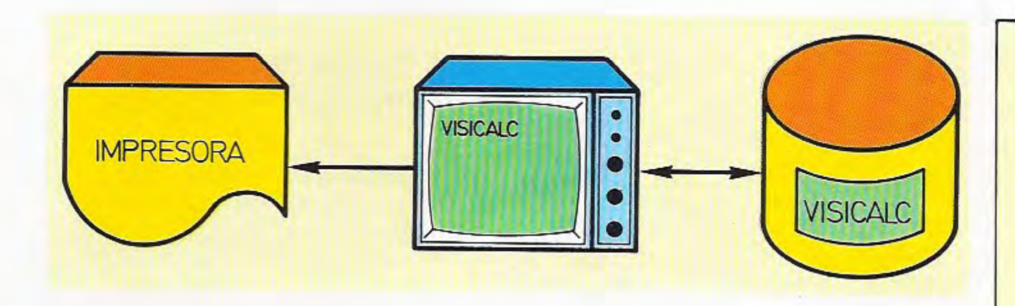
La hoja electrónica Visicalc se creó para permitir simular operaciones financieras, llevar control de costes o comisiones de venta con un ordenador personal. Fue la primera hoja que se creó y está soportada sobre numerosos sistemas operativos, todos los importantes. Para su uso no se requiere ningún conocimiento informático especial. Al ser un programa de los llamados de propósito general, cada usuario define su propia aplicación específica.
Básicamente consiste en una matriz de entrada y salida de datos, ordenada y numerada en filas y columnas, que admite campos alfabéticos, campos numéricos y campos fórmula; estos últimos son los que relacionan los campos numéricos entre sí.
Los comandos para su uso pueden dividirse en tres grandes grupos:

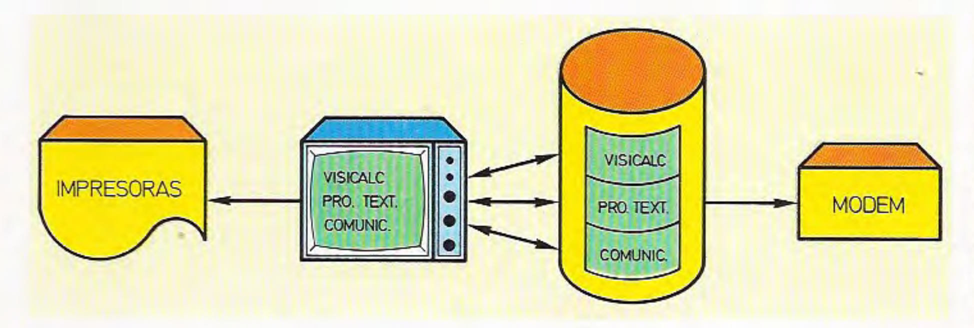
El primero es el grupo de los comandos de preparación de la hoja VisiCalc: órdenes globales del documento, definición de órdenes particulares, borrados e inserción de celdas o casillas, líneas, columnas u hojas completas...
El segundo grupo corresponde a la definición como tal de la aplicación, para lo cual hay que apoyarse en las funciones VisiCalc desarrolladas anteriormente. En este apartado estarán las funciones de movimiento, duplicación de fórmulas, fijación de títulos y creación de ventanas para poder comparar posiciones extremas de la hoja, etc.
El último grupo de comandos es el que trata de las relaciones entre la memoria central del ordenador, donde se ejecuta el programa, y los periféricos del mismo; es decir, la unidad de discos y la impresora.

Queda, por último, el comando referido a la impresora:

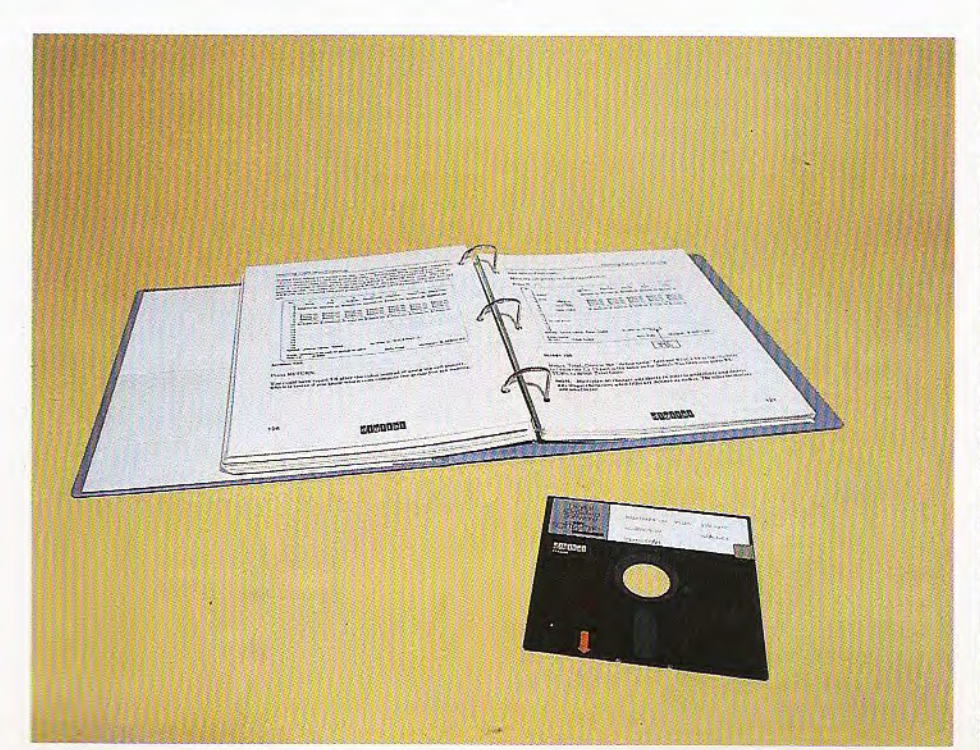
• P: Este comando permite seleccionar el tipo de impresora que se va a utilizar, imprimir un fichero distinto del que tenemos en la memoria central y definir qué posiciones deben imprimirse.

Por ejemplo, si se quiere imprimir una parte de la hoja, desde la posición B8 hasta la H25, habrá que colocar el cursor en la posición origen B-8, activar con /P el comando de impresión y el programa solicitará la última posición. Una vez tecleadas las coordenadas H-25, se empezará a imprimir la zona solicitada.





Inicialmente las hojas electrónicas existían tan solo como aplicación independiente. En la actualidad, son muchos los paquetes integrados que incluyen un entorno de hoja electrónica.



El propio concepto de hoja electrónica nació con el programa VisiCalc.

Para saber más

Prioridad de formatos: globales y parciales

El programa da prioridad a la orden parcial de cada celda sobre la orden global a toda la hoja VisiCalc.
Por ejemplo, si se quieren presentar unas cantidades totales en números enteros y los porcentajes de las mismas con dos decimales, se deberá definir primero toda la hoja en formato entero, con la opción G F I, y luego las casillas parciales de porcentajes como decimales, con los comandos F S.

¿Qué significado tienen los términos absoluto y relativo en el funcionamiento del comando R?

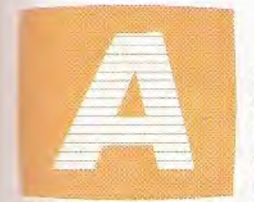
El comando R, de la palabra inglesa REPLICATE, ejecuta una orden de duplicar información. Si lo que se desea es duplicar un campo alfabético o un número concreto, lo duplica sin pedir información adicional (pasa el valor absoluto). Si, por el contrario, deseamos duplicar una campo fórmula, debe indicarse que los valores son relativos. Ejemplo: Si tenemos en la posición B-5 la suma de la zona B-1 a B-4 y se desea que sume también en C-5, de indicar al ordenador que opere con referencias absolutas tendríamos C5 = B1 + B2 + B3 + B4; por el contrario, si se le indica que lo haga con relativas sumará C1 + C2 + C3 + C4.

Ventanas sincronizadas

Al definir las ventanas en VisiCalc, puede establecerse que al mover el cursor se desplace tan sólo la información de una ventana o la de ambas sincronizadamente.

Su majestad el microprocesador

Definición y técnica de funcionamiento



A estas alturas de la obra es ya indiscutible el papel de absoluto protagonismo que representa el microprocesador en la moderna informática. No en balde su nacimiento convulsionó esta disciplina, dando origen a la revolución microinformática.

Su efecto, una década después de ver la luz en los laboratorios de investigación de la firma estadounidense Intel, es más que perceptible en el destacado papel del ordenador personal en la sociedad moderna.

El chip microprocesador —el potente cerebro de los ordenadores personales— va a ser objeto de estudio a lo largo del presente capítulo, en el que se esbozará su definición, características esenciales y método de funcionamiento, tomando como referencia de estudio un modelo de microprocesador experimental.

Definición de microprocesador

La forma más elemental de definir a un microprocesador es como un circuito integrado capaz de ejecutar un programa y controlar las unidades necesarias para dicha ejecución. Como ya se adelantó en el primer tomo de esta obra, las principales aplicaciones del microprocesador son:

— La sustitución de circuitos lógicos, que sólo son utilizables para una única tarea, por circuitos programables capaces de solucionar distintos problemas mediante distintos programas.

— Realizar la labor de unidad central de proceso de un microordenador.

Características de un microprocesador

Los sistemas de microordenadores tendrán unas características muy influenciadas por las del microprocesador en que se basan, ya que tanto su potencia como el resto de sus prestaciones estarán condicionadas por las peculiaridades de su CPU, constituida por el microprocesador.

Veamos cuáles son las principales características de un microprocesador:

- *Longitud de la palabra procesada*

Las longitudes de palabra más comunes de los microprocesadores actuales son 8 y 16 bits, aunque también existen microprocesadores que trabajan con palabras de 32 bits. Cuanto más largas sean las palabras tratadas, mayor será la precisión de cálculo del microprocesador y su capacidad de direccionamiento.

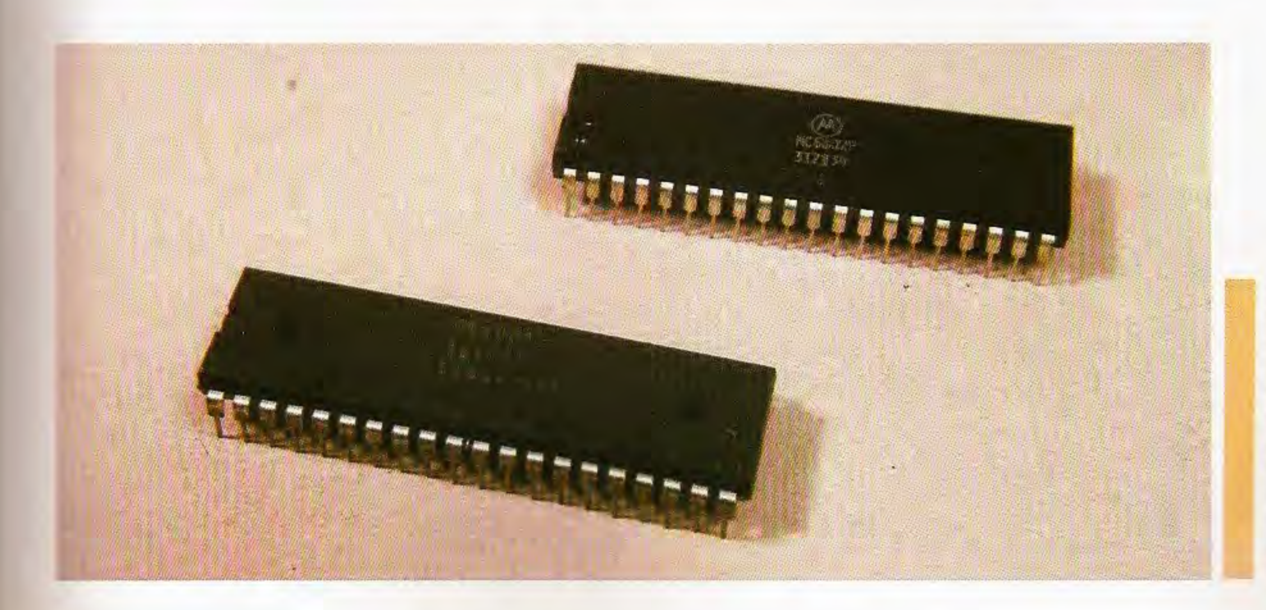
- *Capacidad de memoria*

Esta característica está muy relacionada con la longitud de la palabra procesada. La capacidad máxima de memoria principal accesible por un microprocesador viene marcada por sus posibilidades de direccionamiento. No obstante, microprocesadores de igual longitud de palabra pueden tener distinta memoria en su configuración inicial.

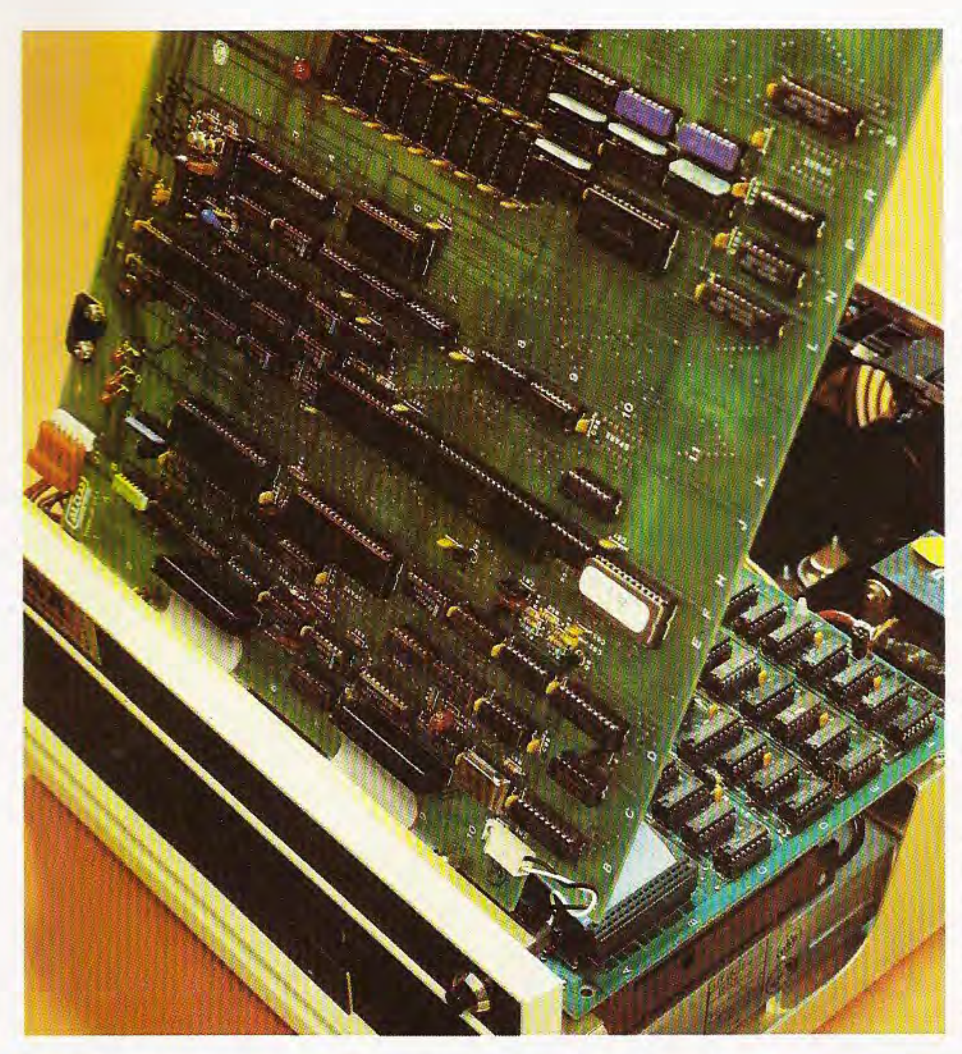
- *Velocidad de ejecución de las instrucciones*

Se denomina ciclo de instrucción al tiempo que invierte el microprocesador en ejecutar completamente una instrucción; con esta característica queda determinada la velocidad de ejecución de un microprocesador. El factor a considerar a la hora de adoptar esta medida como dato característico es que el ciclo difiere según el tipo de instrucción ejecutada.

Otra medida más homogénea es el ciclo de máquina, que refleja el tiempo empleado por el microprocesador en ejecutar una operación elemental de las que se compone cualquier instrucción. Factor determinante es, por supuesto, la frecuencia del reloj que cadencia la actividad del microprocesador. Frecuencias habituales hoy en día son 4,77, 6 u 8 MHz (1 MHz = 1 millón de ciclos por segundo).



El microprocesador es un circuito integrado capaz de ejecutar un programa de instrucciones, controlando a las diversas unidades implicadas en la ejecución.



Aun a pesar de su mínimo volumen, el microprocesador actúa como «cerebro» de potentes microordenadores. Tal es el caso del microprocesador Z-80 que constituye la CPU del sistema multiusuario ALTOS-8010, que aparece en la fotografía.

- *Registros especiales*

Otra característica importante de los microprocesadores es el número de registros especiales que contienen. La mayoría disponen de un único acumulador en la unidad aritmético-lógica; no obstante, existen microprocesadores que incluyen dos acumuladores, con lo que se amplía su potencia y velocidad de operación.

Asimismo, existen dos tendencias en cuanto al resto de los registros internos: una consiste en utilizar parte de la memoria RAM como registros propios del microprocesador; la otra opta por incluir varios registros de trabajo dentro del propio microprocesador.

- *Capacidad de interrupción*

La ejecución de un programa puede ser interrumpida en algunas circunstancias. Una característica básica del microprocesador es la capacidad de recibir y gestionar determinado número de interrupciones.

Mediante estas interrupciones se pueden establecer las comunicaciones necesarias, tanto con el usuario como con otras unidades constitutivas del microordenador, sin que ello afecte a la correcta ejecución del programa en curso.

- *Familia de circuitos complementarios*

La necesidad de complementar la operatividad del microprocesador exige el empleo de una serie de circuitos integrados adaptables al mismo. De esta forma surgen distintas «familias» de circuitos complementarios. Así, por ejemplo, podemos hablar de la familia 6800 o de la 8080 cuando hacemos referencia no sólo a los microprocesadores MOTOROLA-6800 o INTEL-8080, sino también a sus circuitos adaptables.

Actualmente, las familias dominantes son las lideradas por los microprocesadores 8086, 8088, 80286 y 68000.

Programación de un microprocesador

Con todo los visto anteriormente sobre los microprocesadores, podemos llegar a la conclusión de que a nivel físico (hardware) tienen la misma configuración que la unidad central de proceso de un ordenador. Esto no sólo es así como equipo físico, sino que también desde el punto de vista lógico (software) su funcionamiento es análogo.

El microprocesador trabaja directamente en lenguaje máquina; no obstante, si disponemos del correspondiente traductor, podemos utilizar determinados lenguajes de programación más evolucionados. La clasificación de los lenguajes utilizables es la siguiente:

- Lenguaje máquina: Las instrucciones del programa se construyen directamente en binario y son interpretables, sin más, por el microprocesador. En algunos casos, en vez de utilizar un lenguaje binario se puede usar una codificación octal o hexadecimal para su introducción en el ordenador.

Cuando un programa se ha codificado directamente en lenguaje máquina se le suele denominar programa objeto.

- Lenguajes simbólicos: En lugar de recurrir directamente a códigos binarios, como era el caso del lenguaje máquina, se puede trabajar con instrucciones y di-

Nociones de álgebra de Boole (I)

Conceptos primitivos y axiomas

Los conceptos primitivos en que se basa un álgebra de Boole o sistema booleano son los siguientes:

1. *Variable lógica*

Se define como variable lógica a aquella variable que sólo puede tomar dos valores: Verdadero o Falso (que también se puede anotar como: V y F o 1 y 0). Por ejemplo:

$X \equiv$ ¿El viajero desea un billete de clase turista?, es una variable lógica, ya que $X = 1$ o $X = 0$.

2. *Relación de equivalencia (=)*

La relación de equivalencia del álgebra de Boole es la igualdad, que tiene el mismo sentido que la igualdad tradicional. Así, si tenemos dos variables lógicas, X_1 y X_2, la expresión $X_1 = X_2$ implica que ambas variables tienen el mismo valor.

3. *Suma lógica ($\vee$)*

Consiste en una operación sobre variables lógicas que viene definida mediante la siguiente tabla:

X_1	X_2	$X_1 \vee X_2$
0	0	0
0	1	1
1	0	1
1	1	1

También es conocida como operación «O», ya que su funcionamiento es disyuntivo. Por ejemplo, si $X_1 \equiv$ «¿El viajero desea billetes de clase 1.ª?» y $X_2 \equiv$ «¿El viajero desea billetes de clase turista?», entonces $X_1 \vee X_2 \equiv$ «El viajero desea billetes de clase 1.ª *O* de clase turista?», que tendrá respuesta siempre afirmativa, excepto cuando tanto X_1 como X_2 tengan respuesta negativa:

¿Desea 1.ª	¿Desea turista?	¿Desea 1.ª o turista?
0 - NO	0 - NO	0 - NO
0 - NO	1 - SI	1 - SI
1 - SI	0 - NO	1 - SI
1 - SI	1 - SI	1 - SI

4. *Producto lógico ($\wedge$)*

La definición de esta operación booleana se resume en la tabla:

X_1	X_2	$X_1 \wedge X_2$
0	0	0
0	1	0
1	0	0
1	1	1

Al contrario que en el caso anterior, esta operación es de tipo conjuntivo y, por tanto, también se la denomina operación «Y». Por ejemplo, si definimos una nueva variable lógica $X_3 \equiv$ «¿Hay disponibles billetes de 1.ª clase?», entonces $X_1 \wedge X_3 \equiv$ «¿El viajero desea billetes de 1.ª clase *Y* hay billetes disponibles de 1.ª clase?», sólo tendrá respuesta afirmativa cuando ambas variables tomen el valor verdadero:

¿Desea 1.ª	¿Hay 1.ª	¿Desea Y hay 1.ª?
0 - NO	0 - NO	0 - NO
0 - NO	1 - SI	0 - NO
1 - SI	0 - NO	0 - NO
1 - SI	1 - SI	1 - SI

- Si queremos dar una definición formal de sistema booleano (álgebra de Boole), podríamos adoptar la siguiente:

Se dice que un conjunto de variables $S = \{A, B, C, ...\}$ es un sistema booleano si se cumplen las siguientes propiedades:

1. S posee la operación suma lógica ($\vee$).
2. S posee la operación producto lógico ($\wedge$).
3. S posee una relación de equivalencia (=) y se verifican los siguientes axiomas:

1. Para toda variable ($\forall$) A perteneciente ($\in$) a S se cumple que: $A = 0$ ó $A = 1$.
2. $\forall A \in S$: $A = A$.
3. $\forall A, B \in S$: $A = B$ es equivalente ($\Leftrightarrow$) a $B = A$.
4. $\forall A, B, C \in S$: $(A = B) \wedge (B = C)$ implica ($\Rightarrow$) que $A = C$.

5. $\forall A \in S$: $0 \vee A = A$.
6. $\forall A \in S$: $1 \vee A = 1$.
7. $\forall A \in S$: $0 \wedge A = 0$.
8. $\forall A \in S$: $1 \wedge A = A$.

- Se define variable complementaria de A, y se nota $\overline{A}$, a aquella que toma el valor contrario de A, es decir:

$$\overline{A} = \begin{cases} 0 \text{ si } A = 1 \\ 1 \text{ si } A = 0 \end{cases}$$

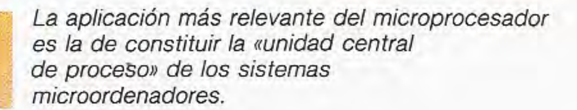
- Se puede demostrar que el resultado de aplicar las operaciones $\vee$, $\wedge$ y $\sim$ (complementación) a variables del sistema es único y, además, la resultante es otra variable del sistema.

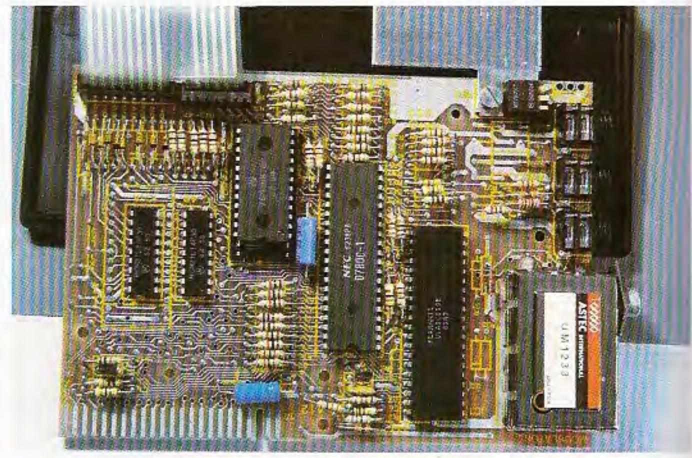
- Si en una expresión booleana verdadera se sustituye $\vee$ por $\wedge$, $\wedge$ por $\vee$, 0 por 1 y 1 por 0, el resultado sigue siendo una expresión booleana verdadera.

FAMILIA MC6800 DE MOTOROLA	
MC6800	Microprocesador.
MC6810A	Memoria RAM estática de 128 bytes.
MC6820	Adaptador de interface para periféricos.
MC6830A	Memoria ROM de 1 Kbyte.
MC6832	Memoria ROM de 2 Kbytes.
MC6840	Temporizador programable.
MC6843	Controlador de disquetes.
MC6845	Controlador de tubo de rayos catódicos.
MC6847	Generador de salidas para vídeo.
MC6850	Interface para comunicaciones asíncronas.
MC6852	Adaptador síncrono para datos serie.
MC6854	Controlador de enlace de datos.
MC6860	Módem digital.
MC6862	Modulador digital.
MC6870	Reloj para microprocesador.
MC6871	Reloj para microprocesador.
MC6880	Cuádruple transmisor bus tri-entrada.
MC6885	Séxtuple buffer tri-estado.
MC68708L	Memoria ROM de 1 Kbyte.
MCM6604L	Memoria RAM dinámica de 4.096 bits.
MPQ6842	Buffer para reloj CPU.

Una de las características del microprocesador se concreta en su familia de circuitos complementarios. En la tabla se relacionan los principales componentes de la familia MC6800.




La aplicación más relevante del microprocesador es la de constituir la «unidad central de proceso» de los sistemas microordenadores.


En su expresión más elemental, un microordenador está constituido por el chip microprocesador (CPU), complementado con algunos circuitos integrados de memoria y de comunicación entrada/salida.

recciones simbólicas. A este tipo de programas se les denomina fuente, y necesitan ser traducidos a lenguaje objeto por un ensamblador.

- Lenguajes de alto nivel: Representan un paso más en la normalización de los lenguajes de programación. Son lenguajes próximos a los hablados convencionalmente y fáciles de interpretar por el usuario.

Al igual que en el caso de los lenguajes simbólicos, los programas redactados en lenguajes de alto nivel reciben el nombre de programas fuente, y antes de ejecutarlos es necesario traducirlos a lenguaje máquina.

Para ilustrar el funcionamiento de un microprocesador vamos a definir en este capítulo un sistema experimental sobre el que se estudiará la distribución de la información, tanto de los datos como de las instrucciones.

Sistema experimental basado en microprocesador

El sistema que vamos a describir constará de cuatro unidades básicas: un periférico de entrada de datos, el microprocesador, una unidad de memoria principal y un periférico de salida de datos. Para estudiar el método operativo las unidades de E/S son irrelevantes, por lo que simplemente consideraremos su existencia sin precisar sus características. En cambio, el microprocesador y la memoria principal funcionarán asociadas, de forma que aquél ejecute los programas y utilice los datos almacenados en ésta. Las características de estas dos unidades, en nuestro sistema experimental, serán las siguientes:

- El microprocesador estará compuesto por las tres zonas básicas ya conocidas: la unidad de control, la unidad aritmético-lógica y un acumulador. La unidad de control se encargará de gobernar y sincronizar todas las operaciones necesarias para la ejecución de los programas.
- Las instrucciones estarán formadas por un código de operación y la dirección de un operando, con el que se realizará el tratamiento indicado por el código de operación. Una vez interpretada la instrucción, la unidad aritmético-lógica se encargará de ejecutarla sobre el registro acumulador. En nuestro sistema tan sólo consideraremos un pequeño subconjunto de posibles instrucciones.

1. Cargar en el acumulador una dato almacenado en memoria.

Código	Operando
CAR	dirección de memoria en la que se encuentra el dato

2. Almacenar el contenido del acumulador en una posición de memoria.

Código	Operando
ALM	dirección de memoria en que se almacenará el contenido del acumulador

3. Sumar al contenido del acumulador un dato almacenado en memoria.

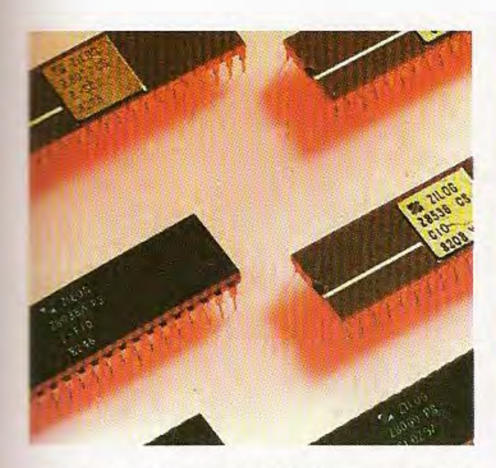
Código	Operando
SUM	dirección de memoria en donde se encuentra el número a sumar

4. Restar del contenido del acumulador un dato almacenado en memoria.

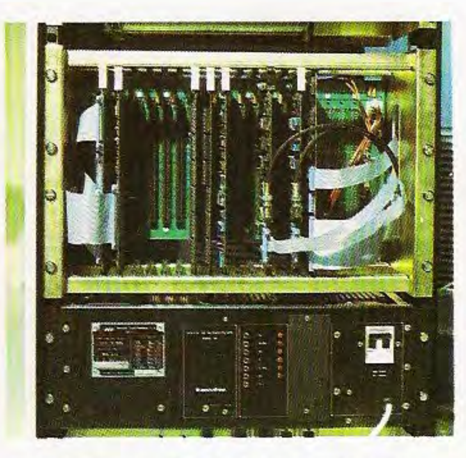
Código	Operando
RES	dirección de memoria en donde se encuentra el número a restar

5. Multiplicar el contenido del acumulador por un dato almacenado en memoria.

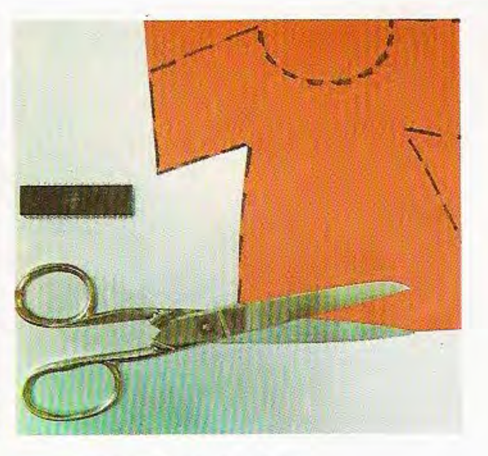
Código	Operando
MUL	dirección de memoria en que se encuentra el número por el que se multiplicará el contenido del acumulador



La existencia de circuitos integrados, especialmente diseñados para complementar la labor del microprocesador, es un factor a considerar a la hora de optar por su empleo como CPU de un microordenador.



La profusión actual de ordenadores de gran potencia operativa y reducido volumen y precio ha sido posible gracias a la existencia del microprocesador: un versátil circuito electrónico programable.



La virtud esencial del microprocesador reside en su plena versatilidad: es el programa de instrucciones quien determina su actuación específica en cada caso.

6. Dividir el contenido del acumulador entre un dato almacenado en memoria.

Código	Operando
DIV	dirección de memoria en que se encuentra el número por el que será dividido el contenido del acumulador

7. Elevar el contenido del acumulador a un dato almacenado en memoria.

Código	Operando
ELE	dirección de memoria en que se encuentra el exponente al que será elevado el contenido del acumulador

Esta instrucción sirve también para la extracción de raíces del contenido del acumulador, ya que:

$$\sqrt[n]{ACUMULADOR} = ACUMULADOR^{1/n}$$



El microprocesador Intel 8088, con arquitectura de 8/16 bits, constituye el «cerebro» de los ordenadores personales adscritos al estándar IBM-PC.

8. Salto incondicional al emplazamiento de una nueva instrucción.

Código	Operando
SAL	dirección de memoria en donde se encuentra la siguiente instrucción

9. Salto condicional: si el contenido del acumulador es distinto de cero, no se ejecutará la próxima instrucción y seguirá la secuencia a partir de la siguiente; si el contenido del acumulador es cero, el programa se ejecutará normalmente.

Código	Operando
CON	—

10. Lectura de un dato a través del periférico de entrada y carga en memoria.

Código	Operando
LEE	dirección de memoria en que se almacenará el dato leído

11. Escritura de un dato a través del periférico de salida.

Código	Operando
ESC	dirección de memoria en que se encuentra almacenado el dato a escribir

12. Fin del programa.

Código	Operando
FIN	—

Todas las operaciones aritméticas se realizan «sobre» el acumulador, es decir, el contenido inicial del acumulador es el primer operando y el contenido final es el resultado de la operación.

• La memoria de nuestro sistema será de tipo RAM y se encargará de almacenar programas y datos. La comunicación se realizará a través de los buses de direcciones, de datos y de control ya estudiados.

Es importante destacar que este sistema experimental no corresponde totalmente a la realidad, si bien es bastante parecido en cuanto al juego de instrucciones y al método de funcionamiento. También debe quedar claro que todas las características esenciales del microprocesador, relacionadas en un párrafo anterior, jugarán un importante papel en el rendimiento global del sistema, pero no alterarán la forma de operar del mismo.

Funcionamiento del sistema

Realizaremos una demostración del funcionamiento del sistema mediante la ejecución de un programa que calcula hipotenusas de triángulos rectángulos en función de los catetos.

El lenguaje que estamos utilizando en esta simulación es de tipo máquina, con

Nociones de álgebra de Boole (II)

Propiedades de un sistema booleano y funciones booleanas

Las propiedades fundamentales de los sistemas booleanos son las siguientes:

1. Ley de idempotente:
$\forall A \in S: A \vee A = A$ y $A \wedge A = A$

2. Ley de involución:
$\forall A \in S: \overline{\overline{A}} = A$ (esto es: el complementario del complementario de una variable lógica es la propia variable).

3. Ley conmutativa:
$\forall A, B \in S: A \vee B = B \vee A$
y $A \wedge B = B \wedge A$

4. Ley asociativa:
$\forall A, B, C \in S: A \vee (B \vee C) =$
$= (A \vee B) \vee C = A \vee B \vee C$ y
$A \wedge (B \wedge C) = (A \wedge B) \wedge C = A \wedge B \wedge C$

5. Ley distributiva:
$\forall A, B, C \in S: A \wedge (B \vee C) =$
$= (A \wedge B) \vee (A \wedge C)$
y $A \vee (B \wedge C) =$
$= (A \vee B) \wedge (A \vee C)$

6. Ley de absorción:
$\forall A, B \in S: A \wedge (B \vee A) = A$
y $A \vee (B \wedge A) = A$

7. Ley de Morgan:
$\forall A, B \in S: \overline{A \vee B} = \overline{A} \wedge \overline{B}$
y $\overline{A \wedge B} = \overline{A} \vee \overline{B}$

El sistema utilizado para demostrar las anteriores proposiciones, y en general cualquier expresión booleana, es siempre el mismo. Se construye una tabla con todas las combinaciones de los posibles valores de las variables que intervengan en la expresión; en cada combinación se calcula el valor correspondiente para los dos términos de la igualdad. La expresión será correcta si en todos los casos los valores coinciden.

Como ejemplo demostraremos la ley de Morgan:

A	B	$\overline{A}$	$\overline{B}$	$A \vee B$	$\overline{A \vee B}$		$\overline{A} \wedge \overline{B}$
0	0	1	1	0	1	=	1
0	1	1	0	1	0	=	0
1	0	0	1	1	0	=	0
1	1	0	0	1	0	=	0

Luego $\overline{A \vee B} = \overline{A} \wedge \overline{B}$,
y análogamente:

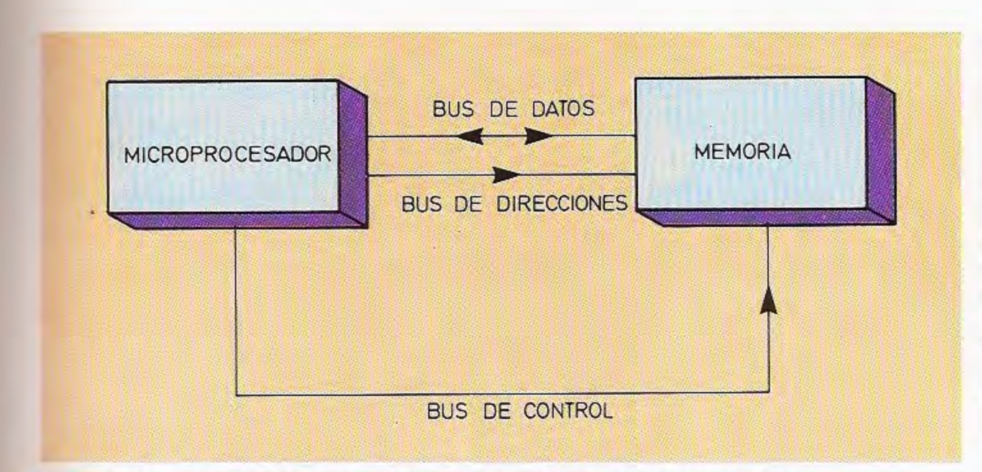
A	B	$\overline{A}$	$\overline{B}$	$A \wedge B$	$\overline{A \wedge B}$		$\overline{A} \vee \overline{B}$
0	0	1	1	0	1	=	1
0	1	1	0	0	1	=	1
1	0	0	1	0	1	=	1
1	1	0	0	1	0	=	0

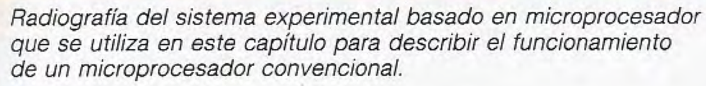
Luego $\overline{A \wedge B} = \overline{A} \vee \overline{B}$, con lo que queda demostrada la ley de Morgan.

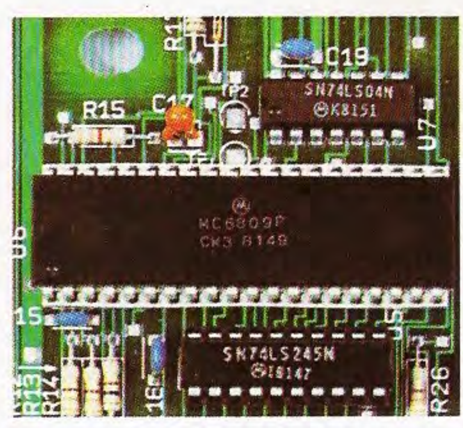
Se llama función booleana a toda aquella que esté definida a partir de variables lógicas sometidas a las operaciones $\vee$, $\wedge$ y $\sim$. Por ejemplo: $f(\overline{X}, Y, Z) = X \wedge (Y \vee \overline{Z})$.

Si asignamos valores concretos a las variables en que se basa una función, podemos calcular el valor de ésta sin más que sustituir las variables por su valor en la expresión. Si, en el ejemplo anterior, $X = 1$ y $Z = 0$, tendremos que:

$f(1,1,0) = \overline{1} \wedge (1 \vee \overline{0}) = 0 \wedge (1 \vee) =$
$= 0 \wedge 1 = 0.$




Radiografía del sistema experimental basado en microprocesador que se utiliza en este capítulo para describir el funcionamiento de un microprocesador convencional.



El microprocesador, en este caso un 6809 de la firma Motorola que desempeña la función de CPU en un THOMSON TO-7, está compuesto por tres zonas básicas: la unidad de control, la unidad aritmético-lógica y el acumulador.

Microprocesadores

Fabricante	Micropro-cesador	Longitud palabra	Ciclo de instrucción mínimo	Ciclo de instrucción máximo	Rango de direccionamiento
INTEL	4040	4 bits	10,8 µseg.	21,6 µseg.	8 K
SIGNETICS	2650	8 bits	1,5 µseg.	6 µseg.	32 K
MOTOROLA	6800	8 bits	1 µseg.	2,5 µseg.	64 K
INTEL	8080	8 bits	1,5 µseg.	3,75 µseg.	64 K
INTEL	8085	8 bits	0,8 µseg.	5,2 µseg.	64 K
ROCKWELL	PPS-8	8 bits	4 µseg.	12 µseg.	16 K
NATIONAL SEMIC.	SC/MPII	8 bits	5 µseg.	10 µseg.	4 K
ZILOG	Z80	8 bits	1 µseg.	5,75 µseg.	64 K
MOTOROLA	68000	16/32 bits	0,5 µseg.	—	16 M
INTEL	8086	8/16 bits	0,4 µseg.	37,8 µseg.	1 M
INTEL	8088	8/16 bits	0,4 µseg.	—	1 M
INTEL	80286	8/16 bits	—	—	16 M
NATIONAL SEMIC.	PACE	16 bits	2,5 µseg.	5 µseg.	64 K
ZILOG	Z8000	16 bits	0,75 µseg.	90 µseg.	48 M

Tabla con las características principales de los microprocesadores más usuales (1 K = 1.024 bytes, 1 M = 1.048.576 bytes).

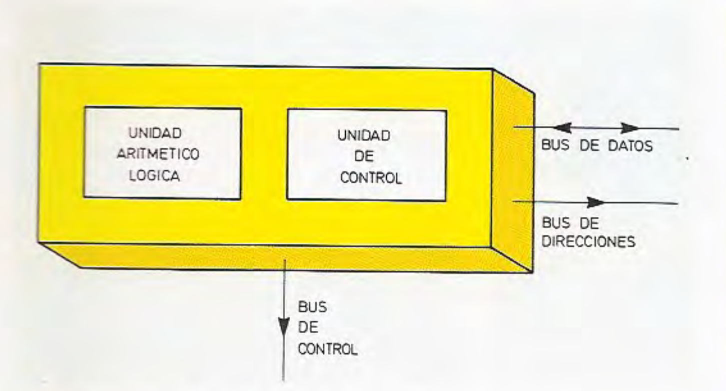
Los microprocesadores actuales suelen ser, comúnmente, de 8 ó 16 bits. En la tabla se relacionan algunos de los microprocesadores más utilizados a lo largo de la escasa década de vida de estos «cerebros» integrados.

la salvedad de que representamos los códigos de operación por medio de símbolos nemotécnicos y trabajamos con el sistema de numeración decimal.

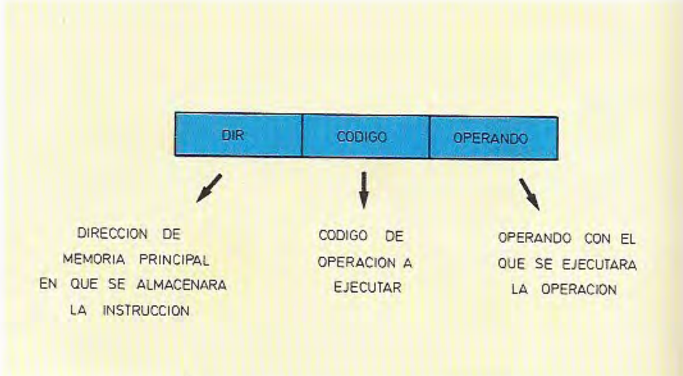
Un programa que realice el cálculo de la hipotenusa de un triángulo rectángulo a partir de los dos catetos podría ser el siguiente:

Dirección de la instrucción	Código de operación	Operando
00	LEE	16
01	LEE	17
02	LEE	18
03	CAR	17
04	CON	—
05	SAL	15
06	MUL	17
07	ALM	19
08	CAR	18
09	MUL	18
10	SUM	19
11	ELE	16
12	ALM	19
13	ESC	19
14	SAL	01
15	FIN	—

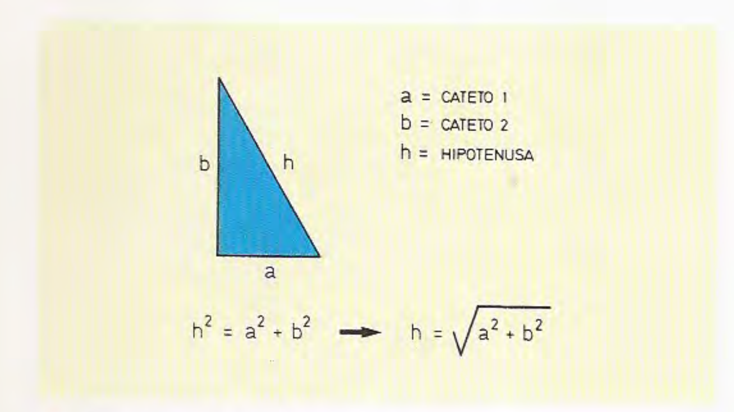
El número de instrucciones necesarias para programar el tratamiento adecuado de los datos es 16; por tanto, se ocuparán 16 palabras (desde la 0 hasta la 15) de la memoria principal para almacenar el programa.



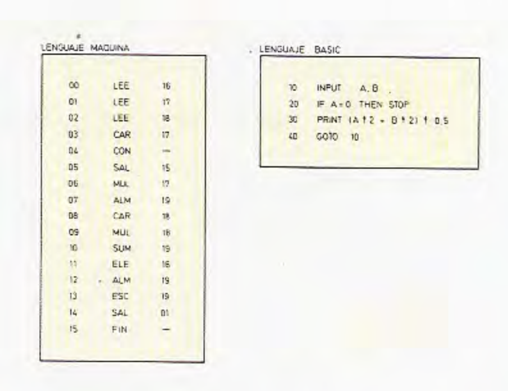
El gráfico muestra la estructura interna de nuestro microprocesador experimental. Consta de: unidad de control, unidad aritmético-lógica con el registro acumulador, bus de datos, direcciones y control.



Formato general de las instrucciones del microprocesador experimental. La casilla de la izquierda refleja la dirección de la memoria principal, la central el código de operaciones y la tercera el operando de la instrucción.



El programa ejemplo aplica el teorema de Pitágoras para encontrar el valor de la hipotenusa de un triángulo rectángulo, conocidos los valores de los catetos.



Dos versiones del programa ejemplo: escrito en lenguaje máquina (izquierda) y en lenguaje BASIC (derecha). La comodidad que supone para el programador el empleo de un lenguaje de alto nivel se hace patente al observar ambos listados.

Se han elegido las posiciones 16 a 19 de memoria para los datos utilizados por el programa, aunque se podía haber tomado otra zona de memoria no consecutiva al programa.

El funcionamiento del microprocesador al ejecutar el anterior programa será el siguiente:

— La primera instrucción tratada será la [00 | LEE | 16], con lo que la unidad de control leerá un dato a través del periférico y lo cargará en la posición 16 de la memoria principal. Este dato debe ser una constante de valor 0,5 y será utilizada para extraer las raíces cuadradas.

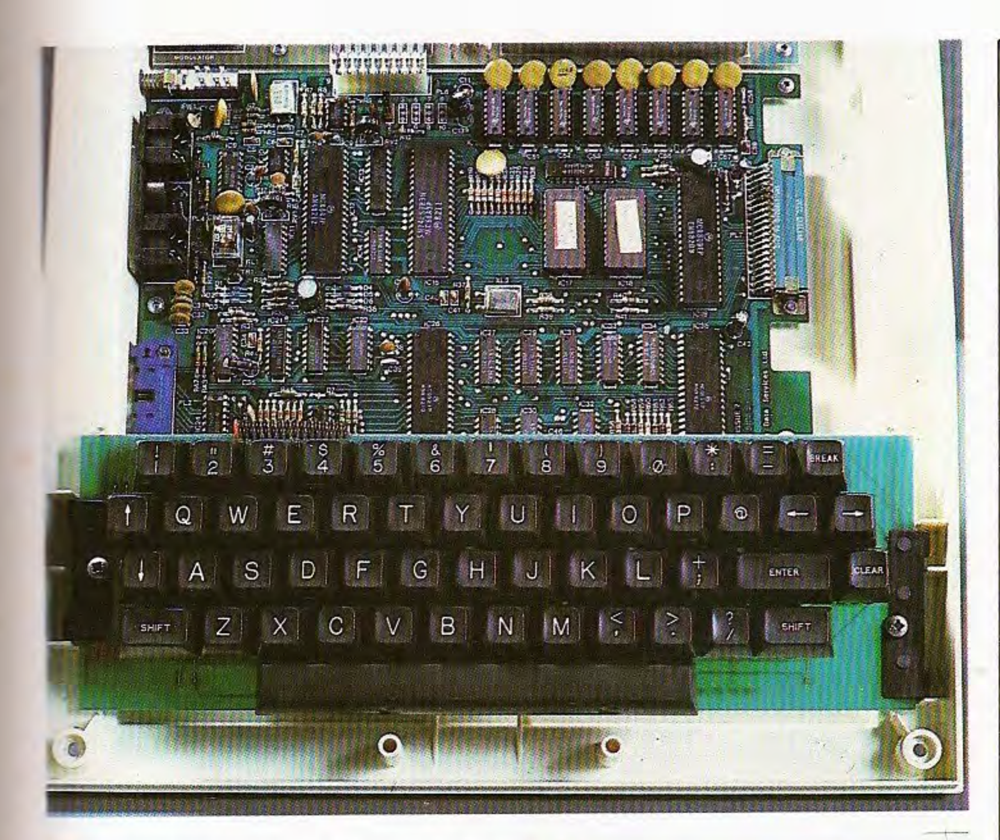
— A continuación, siguiendo la secuencia del programa, se ejecuta [01 | LEE | 17]. En ese momento la unidad de control admitirá por el periférico de entrada el valor del primer cateto y lo cargará en la posición de memoria 17.

Por medio de la instrucción [02 | LEE | 18] se realiza la misma operación con el segundo cateto, que quedará cargado en la posición de memoria 18.

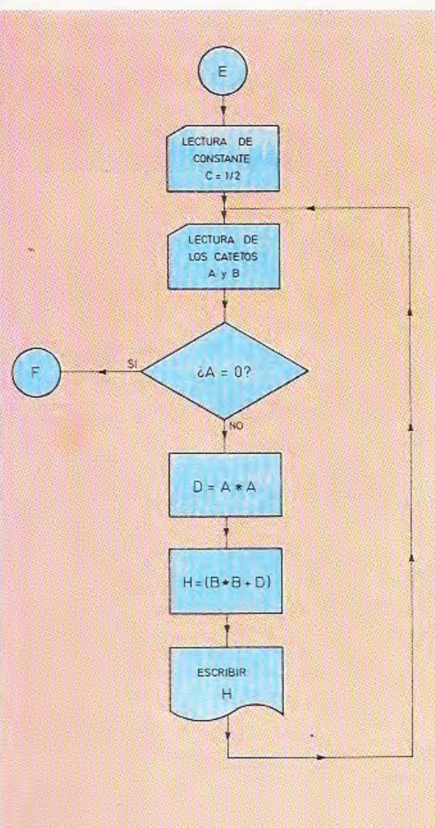
— La instrucción [03 | CAR | 17] se limita a cargar el contenido de la posición 17 de la memoria (primer cateto) en el acumulador.

— Una vez realizada la carga anterior, el programa ejecutará la instrucción de salto condicional [04 | CON | —]: si el contenido del acumulador es cero, se ejecutará la siguiente instrucción [05 | SAL | 15], y si es distinto de cero, ésta no será ejecutada, continuando directamente en la instrucción [06 | MUL | 17].

Como el contenido del acumulador al ejecutar esta instrucción coincide con el primer cateto, el programa sólo «saltará» a la instrucción [15 | FIN | —]



Circuitería electrónica constitutiva de la unidad central del ordenador doméstico Dragon-32. El elemento esencial es el microprocesador Motorola 6809.



La figura representa el algoritmo utilizado para resolver el problema planteado: cálculo de la hipotenusa de un triángulo rectángulo.

Para saber más

¿Existe alguna diferencia en la forma de representar una instrucción y un dato?

En principio, no. Cualquier byte de la memoria principal puede representar indistintamente una instrucción o un dato; será el programador, a través del programa, el que determine su distribución.

¿Qué parte del microprocesador se encarga de ejecutar las instrucciones?

La unidad de control examina el código de operación de la instrucción a ejecutar y, según sea éste, realiza ella misma la operación deseada u ordena su ejecución a la unidad aritmético-lógica.

En la instrucción de bifurcación o salto condicional, ¿cuándo se ejecuta la siguiente instrucción y cuándo se salta?

Según el valor que contenga el acumulador, se saltará o no a la siguiente instrucción. En el juego de instrucciones definido para el sistema experimental, la próxima instrucción se ejecuta cuando el contenido del acumulador es distinto de cero y se salta en caso contrario. No obstante, en algunos sistemas el proceso es al contrario.

¿De qué forma controla el programador los datos cargados en la memoria principal?

Al emplear lenguajes de máquina, el programador tiene que saber en todo momento la dirección de memoria en la que se almacena el dato; sin embargo, al utilizar lenguajes simbólicos o de alto nivel, se limita a darles un nombre nemotécnico y el propio sistema decide la dirección. Por ejemplo, en vez de inscribir la instrucción | 02 | LEE | 18 | podrá hacerlo, por tanto, en la forma: | 02 | LEE | CATETO 2 | ; en cualquier otra parte del programa en la que necesite el valor del segundo cateto, sólo tendrá que escribir CATETO 2 y el microprocesador buscará en la dirección apropiada.

cuando éste sea cero, es decir: el programa calculará tantas hipotenusas como pares de catetos se le suministren y sólo terminará cuando se le proporcione una pareja de catetos de valor cero.

— Si la anterior instrucción no ha provocado la conclusión del programa, al ejecutar | 06 | MUL | 17 | la unidad aritmético-lógica multiplica el contenido del acumulador (primer cateto) por el contenido de la posición de memoria 17 (primer cateto), almacenando el resultado (primer cateto al cuadrado) en el acumulador.

— La siguiente instrucción | 07 | ALM | 19 | llevará el contenido del acumulador a la posición de memoria 19. El hecho de almacenar el primer cateto al cuadrado en una posición intermedia de memoria es debido a que será utilizado posteriormente.

— A continuación se ejecuta la instrucción | 08 | CAR | 18 |, con la que se cargará la posición de memoria 18 (segundo cateto) en el acumulador. Si en la instrucción anterior no se hubiera almacenado el valor del primer cateto al cuadrado en una posición intermedia, ahora se habría perdido este dato.

Mediante la instrucción | 09 | MUL | 18 | la unidad aritmético-lógica multiplicará el contenido de la posición de memoria 18 (segundo cateto) por el contenido del acumulador (segundo cateto) y almacenará el resultado (segundo cateto al cuadrado) en el acumulador.

— La instrucción | 10 | SUM | 19 | obtendrá la suma de la posición 19 de la memoria principal (primer cateto al cuadrado) y el contenido del acumulador (segundo cateto al cuadrado) y almacenará el resultado (suma de los cuadrados de los catetos) en el acumulador.

— Por último, la instrucción | 11 | ELE | 16 | eleva el contenido del acumulador al número contenido en la posición 16 de la memoria principal. Como dicho número es 1/2 = 0,5 (cargado mediante la instrucción | 00 | LEE | 16 |), el resultado obtenido es la raíz cuadrada de la suma de los cuadrados de los catetos, con lo que ya tenemos el resultado definitivo cargado en el acumulador.

— Para poder escribir el valor de la hipotenusa a través del periférico de salida, primero es necesario almacenar dicho valor en una posición de memoria; esto se consigue con la instrucción | 12 | ALM | 19 |, que almacenará el contenido del acumulador en la posición 19 de la memoria principal.

— A continuación se ejecuta la instrucción | 13 | ESC | 19 |, con lo que se obtendrá el listado (si el periférico de salida es una impresora) del valor de la hipotenusa.

— La instrucción | 14 | SAL | 01 | origina un salto incondicional a la instrucción | 01 | LEE | 17 |, y se repetirá todo el proceso anteriormente descrito.

— Si en la instrucción | 05 | SAL | 15 | se produjo el salto a la instrucción | 15 | FIN | – |, al ser ejecutada ésta se dará por terminado el trabajo, con lo que el microprocesador quedará libre.



Aunque resulte sorprendente, los modernos y potentes microordenadores no son más que una aplicación directa del microprocesador.

Para saber más

¿Qué es un microprocesador: un circuito integrado o un ordenador?

Un microprocesador es básicamente un circuito integrado, que puede utilizarse como circuito lógico programable o como unidad central de proceso de un microordenador. No se le puede considerar como un ordenador, aunque sí constituye su «cerebro».

¿Cuáles son las principales características de un microprocesador:

- La longitud de palabra procesada.
- La capacidad de memoria.
- La velocidad en la ejecución de instrucciones.
- Los registros de que dispone.
- La capacidad que tiene para interrumpir programas en ejecución.
- La familia de circuitos complementarios que se le pueden adaptar.

¿La memoria forma parte de la estructura interna del microprocesador?

No. Las unidades de memoria principal no están integradas en el microprocesador, pero el espacio máximo de memoria direccionable depende de su capacidad de direccionamiento.

¿Cómo se puede programar un microprocesador?

Se programa por medio de secuencias de instrucciones o programas. Las instrucciones deben estar codificadas en un lenguaje inteligible por el microprocesador: el lenguaje máquina.

La información en el microprocesador

Ejecución de las instrucciones



Un microprocesador trata dos tipos de informaciones: las instrucciones del programa, que serán interpretadas por la unidad de control, y los datos, de cuyo tratamiento se ocupa la unidad aritmético-lógica de acuerdo con lo indicado por las instrucciones. En este capítulo analizaremos el flujo de información que se establece dentro del microprocesador cuando éste ejecuta una instrucción básica de su repertorio interno.

El gráfico muestra el flujo interno de información dentro de un sistema basado en microprocesador. Los códigos de operación son interpretados por la unidad de control, mientras que los operandos son tratados por la unidad aritmético-lógico.

Ejecución secuencial

Las instrucciones a ejecutar por el microprocesador constan de dos zonas: *código de operación y operando* (uno o varios).

Por ejemplo, en la operación de adición de dos números (A + B), A y B son los operandos (en este caso los operandos coinciden con los datos reales) y el signo «+» es el código de operación. Del tratamiento de cada una de las zonas que componen la instrucción se ocupan distintas unidades internas del microprocesador. Así, los *códigos de operación* son interpretados por la unidad de control, mientras que los *operandos* son tratados por la unidad aritmético-lógica.

Para dirigir de forma secuencial las palabras binarias de información hacia la unidad de control o hacia la unidad aritmético-lógica, y de esta forma lograr la correcta ejecución del programa, es preciso conocer perfectamente las posiciones de memoria en las que reside tal información.

Como ya sabemos, las diversas posiciones de memoria están identificadas por una *dirección* específica. Para acceder a la posición de memoria adecuada en cada instante, el microprocesador cuenta con un registro especializado en esta tarea, denominado «contador de programas» o «contador de direcciones»; en cada instante, el contenido del citado registro coincide con la dirección correspondiente a la posición de memoria a la que accederá el microprocesador, bien sea para extraer o para almacenar en ella una palabra de información.

Por el momento, nuestro interés se centra en conocer el método que sigue el microprocesador para ejecutar un programa o conjunto de instrucciones integradas por códigos de operación (órdenes) y operandos (datos o direcciones correspondientes al emplazamiento en memoria de los datos afectados).

El proceso de ejecución empieza con la lectura por parte del microprocesador del contenido de una posición de memoria, posición de memoria que, por supuesto, será direccionada por el contador de programas. Para que la ejecución se desarrolle de forma automática a partir de este preciso instante, es necesario que la primera palabra extraída de memoria coincida con un «código de operación», puesto que al ser ésta una orden indicará al microprocesador cuál es la naturaleza (código de operación o dato) de la próxima palabra a la que accederá.

Pero volvamos a ocuparnos de la primera palabra de información que se extrae de la memoria. Esta ingresa en el microprocesador a través del denominado «bus de datos» y pasa al interior de un registro denominado «registro de instrucciones».

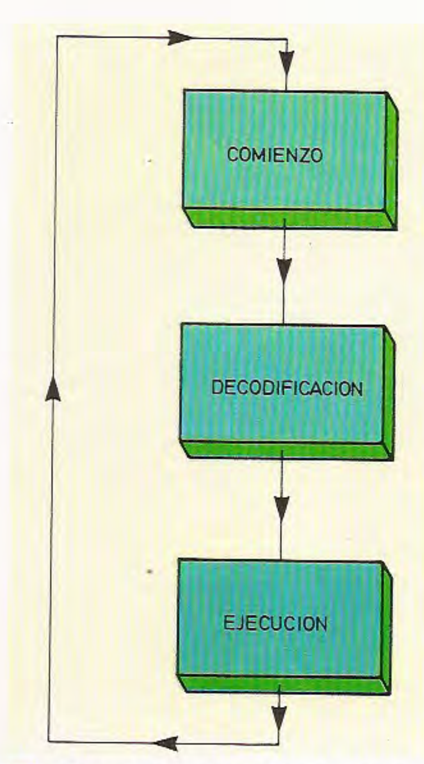
Acto seguido, la unidad de control interpretará la citada palabra que, como sabemos, coincide con un código de operación u orden a ejecutar. A raíz de su interpretación, la unidad de control conocerá qué operación debe ejecutar y cuál será la naturaleza de la próxima palabra que debe leer en la memoria. Si la nueva palabra es un código de operación, deberá aplicarle un tratamiento de interpretación análogo al descrito; no obstante, si se trata de un operando lo dirigirá hacia la unidad aritmético-lógica y generará las órdenes necesarias para que reciba el adecuado tratamiento.

El flujo de información se establece a través de los tres buses que ya estudiamos anteriormente:

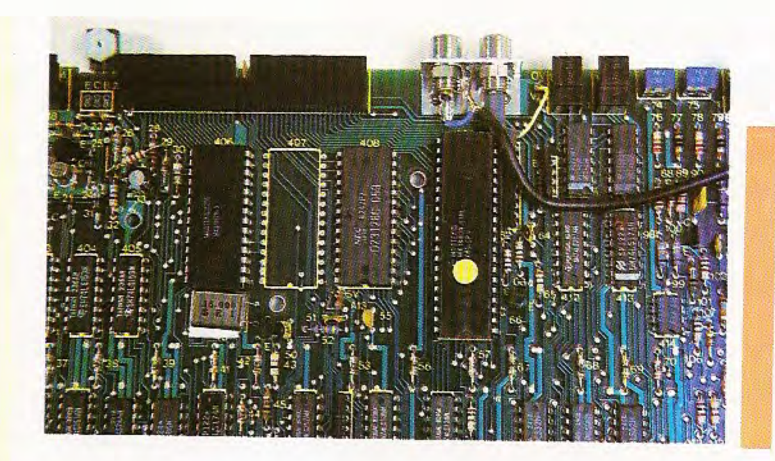
— El *bus de datos,* que se encargará de «traer» y «llevar» la información desde o hacia la memoria.

— El *bus de direcciones,* que «apuntará» (direccionará) la posición de memoria de la que se leerá o en la que se escribirá la palabra de información.

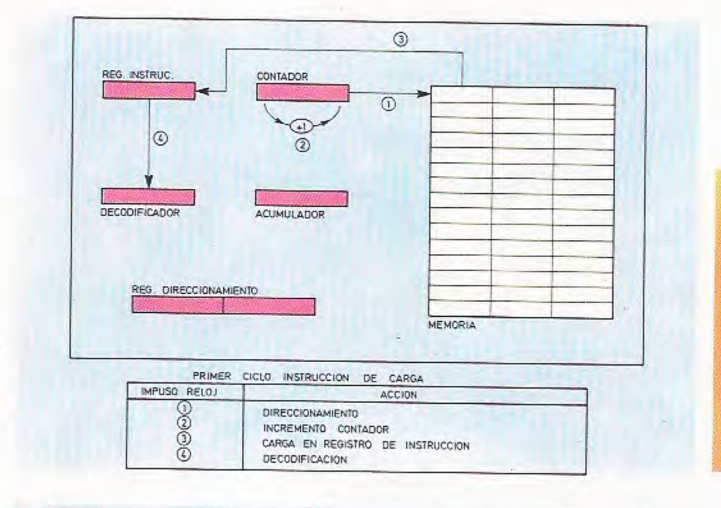
— El *bus de control,* que transmitirá las órdenes (microinstrucciones) generadas a raíz de la interpretación de los códigos de operación de las instrucciones.



Estas son las tres fases de que consta el tratamiento de una instrucción por parte del microprocesador: comienzo, decodificación y ejecución.



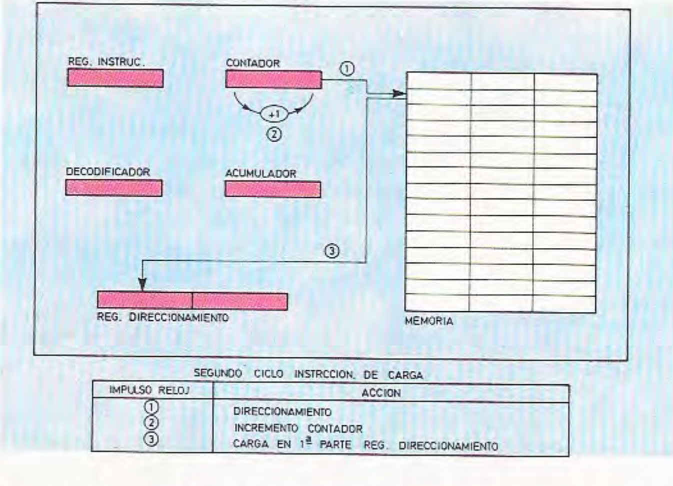
El microprocesador es el elemento central de los sistemas microordenadores: procesa los datos recibidos en virtud de lo ordenado por las instrucciones del programa.



PRIMER CICLO INSTRUCCION DE CARGA

IMPUSO RELOJ	ACCION
①	DIRECCIONAMIENTO
②	INCREMENTO CONTADOR
③	CARGA EN REGISTRO DE INSTRUCCION
④	DECODIFICACION

Durante el primer ciclo de la ejecución de una instrucción de carga, el microprocesador se encarga de recoger y decodificar el código de operación.



SEGUNDO CICLO INSTRCCION. DE CARGA

IMPULSO RELOJ	ACCION
①	DIRECCIONAMIENTO
②	INCREMENTO CONTADOR
③	CARGA EN 1ª PARTE REG. DIRECCIONAMIENTO

En el segundo ciclo de la instrucción de carga, el microprocesador deposita los ocho primeros bits del operando —dirección del dato— en la primera parte del registro de direccionamiento.

Ejecución de una instrucción típica

El proceso de ejecución se inicia llevando al bus de direcciones el contenido del registro contador de direcciones, con lo que se apuntará a la posición de memoria que contiene la primera zona de la instrucción: el código de operación.

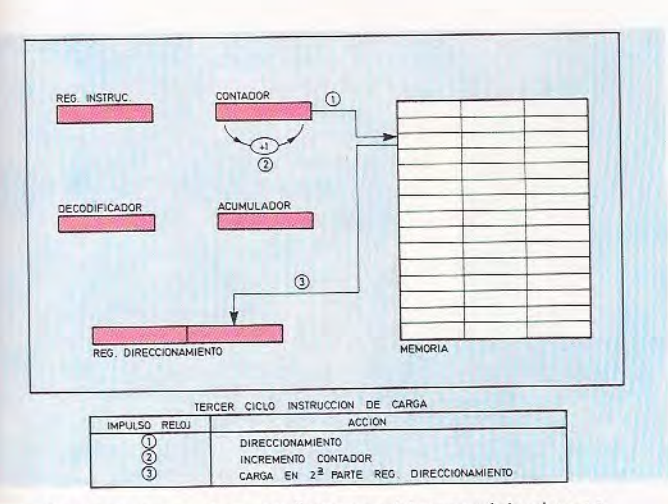
A través del bus de control, el microprocesador indicará a la unidad de memoria que desea «leer» el contenido de la posición direccionada; la referida palabra de información pasará al bus de datos y éste al interior del registro de instrucciones del microprocesador. Esta primera fase en el tratamiento de la instrucción se denomina «fase inicial o de comienzo».

De este punto arranca la segunda fase —«decodificación»—, en el transcurso de la cual la unidad de control decodifica e interpreta el código de operación depositado en el registro de instrucciones.

Por último, la unidad de control generará las órdenes adecuadas dentro de la denominada «fase de ejecución».

Si la ejecución de la instrucción precisa de algunos bits, la unidad de control se encargará de proporcionárselos y, posteriormente, se pasará a ejecutar una nueva instrucción.

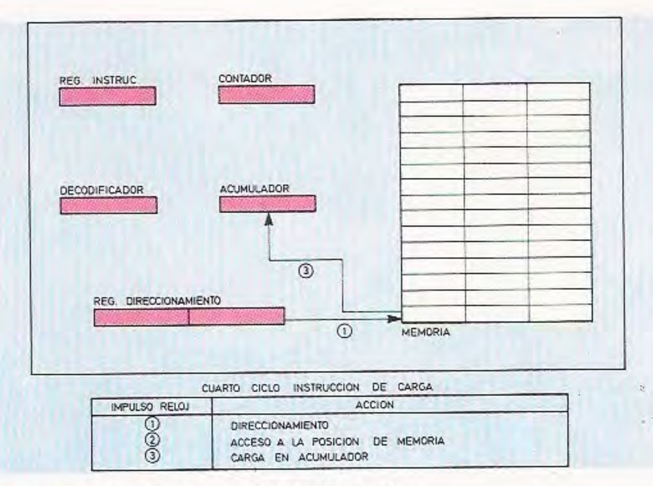
Cada vez que se accede a la memoria (fase inicial) empieza lo que se denomina un ciclo de máquina, que contiene de dos a cinco estados internos, cada uno



TERCER CICLO INSTRUCCION DE CARGA

IMPULSO RELOJ	ACCION
①	DIRECCIONAMIENTO
②	INCREMENTO CONTADOR
③	CARGA EN 2ª PARTE REG. DIRECCIONAMIENTO

Operaciones que se realizan en el tercer ciclo de una instrucción de carga. Su desarrollo es análogo al del segundo ciclo, con la diferencia de que la palabra binaria se carga en la segunda parte del registro de direccionamiento.



CUARTO CICLO INSTRUCCION DE CARGA

IMPULSO RELOJ	ACCION
①	DIRECCIONAMIENTO
②	ACCESO A LA POSICION DE MEMORIA
③	CARGA EN ACUMULADOR

En el cuarto ciclo de la instrucción de carga se efectúa la lectura o carga del dato real en el acumulador. La dirección del dato viene dada por el operando.

Nociones de álgebra de Boole (y III)

Minimización de funciones booleanas

Como ya vimos anteriormente, se pueden definir funciones de variables booleanas $f(A_1\ A_2, \ldots, A_n)$ que serán muy útiles para el diseño de circuitos. Es importante que la expresión utilizada sea la mínima: por ejemplo, $f_1(A_1, \overline{A_2}) = A_1 \vee A_2$ y $f_2(A_1, A_2) = A_1 \vee (\overline{A_1} \wedge \overline{A_2})$ son la misma función, ya que para todos los posibles valores de las variables dan el mismo resultado:

$A_1\ A_2$	$\overline{A_1}\ \overline{A_2}$	$\overline{A_1} \wedge \overline{A_2}$	$A_1 \vee \overline{A_2}$		$A_1 \vee (\overline{A_1} \wedge \overline{A_2})$
0 0	1 1	0	1	=	1
0 1	1 0	0	0	=	0
1 0	0 1	0	1	=	1
1 1	0 0	0	1	=	1

Sin embargo, a la hora de implementar un circuito basado en una de las dos funciones será más sencillo utilizar f_1 que f_2. En el caso del ejemplo, de f_2 a f_1 se puede pasar «a simple vista» sin más que aplicar algunas de las propiedades estudiadas, pero en casos más complicados, para llegar a la minimización es necesario utilizar un procedimiento más meticuloso.

Funciones booleanas en forma canónica de «minterms» o «maxterms»

Dada una función booleana $f(A_1, \ldots, A_n)$, un *minterm* relativo a la función f es una expresión que agrupa a todas las variables, bien en su forma normal o bien en su forma complementaria, relacionándolas por medio de la operación producto lógico. Se puede demostrar que toda función booleana admite una única forma canónica de disyunciones de *minterms*. Análogamente se podría definir *maxterm* sin más que cambiar «producto lógico» por «suma lógica» y el teorema enunciado seguiría siendo válido.

Un procedimiento para obtener la forma canónica de *minterms* de una función booleana puede ser el siguiente:

Se representan en una tabla todas las posibles combinaciones de valores para las diversas variables y se calcula el valor de la función para cada una de ellas. Una vez realizada esta operación, se toman los *minterms* de los resultados verdaderos, haciendo corresponder las variables en su forma normal a las que tomaron inicialmente valor uno y en su forma complementaria a las que tomaron valor cero. Por ejemplo, para poner en forma canónica de *minterms* $f(A_1, A_2, A_3) = [(A_1 \vee \overline{A_2}) \vee A_3] \wedge [(\overline{A_1} \wedge \overline{A_2}) \vee A_3]$ empezaremos construyendo la siguiente tabla:

$A_1\ A_2\ A_3$	*Minterm*	$f(A_1, A_2, A_3)$	Interviene
0 0 0	$\overline{A_1} \wedge \overline{A_2} \wedge \overline{A_3}$	0	NO
0 0 1	$\overline{A_1} \wedge \overline{A_2} \wedge A_3$	1	SI
0 1 0	$\overline{A_1} \wedge A_2 \wedge \overline{A_3}$	0	NO
0 1 1	$\overline{A_1} \wedge A_2 \wedge A_3$	0	NO
1 0 0	$A_1 \wedge \overline{A_2} \wedge \overline{A_3}$	0	NO
1 0 1	$A_1 \wedge \overline{A_2} \wedge A_3$	1	SI
1 1 0	$A_1 \wedge A_2 \wedge \overline{A_3}$	0	NO
1 1 1	$A_1 \wedge A_2 \wedge A_3$	1	SI

Luego su forma canónica en *minterms* es:
$f(A_1, A_2, A_3) = (\overline{A_1} \wedge \overline{A_2} \wedge A_3) \vee (A_1 \wedge \overline{A_2} \wedge A_3) \vee (A_1 \wedge A_2 \wedge A_3)$

Procedimiento de minimización

Para minimizar una función booleana (expresarla de la forma más simple) se pueden seguir los siguientes pasos:

1. Obtener su representación canónica en forma de *minterms* o *maxterms*.
2. Obtener los bloques principales de *minterms* o *maxterms* de la matriz de Karnaugh.
3. Seleccionar el recubrimiento mínimo.

A título de ejemplo, y sin hacer una descripción formal de cada uno de los puntos, vamos a obtener la minimización de:
$f(A_1, A_2, A_3) = [(A_1 \vee \overline{A_2}) \wedge A_3] \wedge [(\overline{A_1} \wedge \overline{A_2}) \vee A_3]$

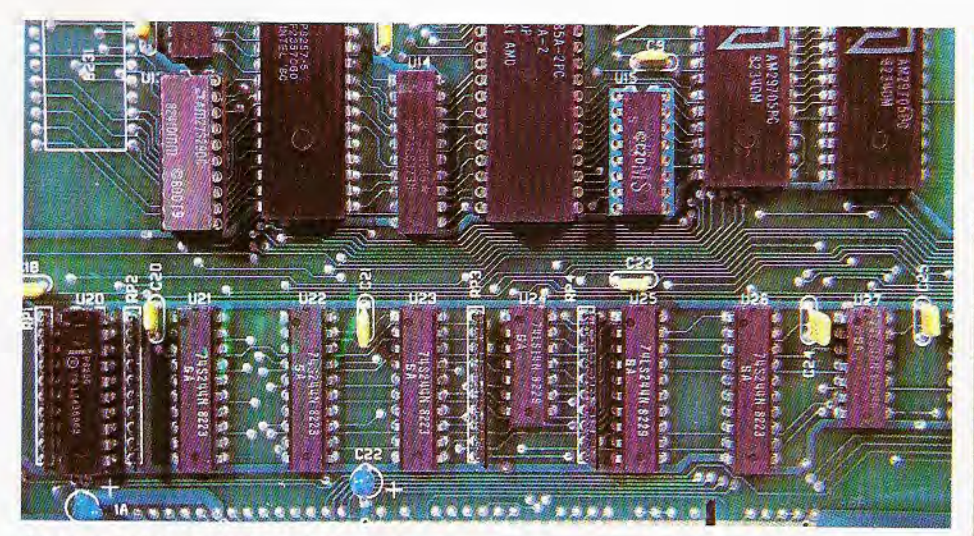
1. Su forma canónica de *minterms* la obtuvimos en el ejemplo anterior:
$f(A_1, A_2, A_3) = \overline{A_1} \wedge \overline{A_2} \wedge A_3 \vee (A_1 \wedge \overline{A_2} \wedge A_3) \vee (A_1 \wedge A_2 \wedge A_3)$
2. La matriz de Karnaugh será:

	A_2		$\overline{A_2}$	
A_1	0	1	1	0
$\overline{A_1}$	0	0	1	0
	$\overline{A_3}$	A_3		$\overline{A_3}$

3. Y un recubrimiento mínimo de la anterior matriz es:

	A_2		$\overline{A_2}$	
A_1	0	1	1	0
$\overline{A_1}$	0	0	1	0
	$\overline{A_3}$	A_3		$\overline{A_3}$

Con lo que $f(A_1, A_2, A_3) = (A_1 \wedge A_3 \vee (A_2 \wedge A_3)$ es la expresión más simple posible de la función original.



El microprocesador no sólo es capaz de procesar los datos que se le suministran, sino que también cuenta entre sus funciones la de controlar a las unidades implicadas en el tratamiento.

de los cuales corresponderá a una microinstrucción.

Ejecución de una instrucción específica

Con el fin de estudiar el flujo de información y las señales generadas por la unidad de control del microprocesador, vamos a describir el proceso de ejecución de una instrucción de carga. Para facilitar la descripción supondremos que se trata de un microprocesador (en nuestro caso experimental) de 8 bits, dotado de un bus de direcciones de 16 bits.

Instrucción de carga o lectura

Lee el contenido de la posición de memoria direccionada y lo carga en el acumulador. Esta instrucción se desarrolla normalmente en cuatro ciclos de máquina.

- Primer ciclo de máquina: Se encarga de recoger y decodificar el código de operación. Con el primer impulso del reloj, el contenido del contador de programas es colocado sobre el bus de direcciones.

Con el segundo impulso del reloj, a través del bus de control se envía una orden de lectura y se suma una unidad al contador de programas.

Con el tercer impulso del reloj, el contenido de la posición de memoria es enviado al registro de instrucciones a través del bus de datos.

Después del cuarto impulso del reloj, el contenido del registro de instrucciones será decodificado. La configuración binaria del código de operación indica que las dos siguientes posiciones de memoria contienen el operando (de 16 bits), que coincide con la dirección en la que se encuentra el dato implicado en la ejecución de la instrucción).

- Segundo ciclo de máquina: Carga de los primeros ocho bits del operando (dirección de dato) en la primera parte del registro de direccionamiento.

De nuevo se efectúa una operación de lectura en memoria, aunque depositando la palabra binaria leída en la primera parte del registro de direccionamiento.

- Tercer ciclo de máquina: Carga del segundo byte del operando (dirección del dato) en la segunda parte del registro de direccionamiento.

Se desarrolla de forma análoga al segundo ciclo de máquina, exceptuando que la palabra binaria se carga en la segunda parte del registro de direccionamiento.

- Cuarto ciclo de máquina: Durante el mismo se realiza la carga del dato real (cuya dirección la aportaba el operando) en el acumulador.

Con el primer impulso de reloj se envía el contenido del registro de direccionamiento al bus de direcciones.

El segundo impulso coincide con la lectura del dato y su traslado al bus de datos.

Con el tercer impulso de reloj, la información contenida en el bus de datos es transferida al acumulador; al mismo tiempo se pone a cero el registro de instrucciones, que queda libre para recibir la siguiente instrucción del programa.



Para saber más

¿De cuántas fases consta la ejecución de una instrucción?

De tres. La primera se denomina de «comienzo» y consiste en depositar el código de operación de la instrucción en el registro de instrucciones, a través del bus de datos. La segunda es la fase de «decodificación», que se encarga de decodificar e interpretar el código de operación, y la tercera es la fase de «ejecución» propiamente dicha.

¿Qué es un ciclo de máquina?

Es un conjunto de microinstrucciones, generalmente de dos a cinco, que permite ejecutar parte de una instrucción.

¿Cuándo se dice que un microordenador tiene «funcionamiento síncrono»?

Cuando las microinstrucciones son generadas al mismo ritmo con el que se emiten los impulsos de reloj. La mayoría de los microordenadores tienen funcionamiento síncrono.

Si el tamaño de la palabra procesable por el microordenador es de 8 bits y el direccionamiento se realiza mediante 16 bits, ¿cómo se pueden tratar estos 16 bits simultáneamente?

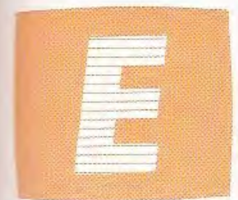
En ese caso el registro de direccionamiento contiene 16 bits y el bus de datos sólo 8; por tanto, la transmisión se realiza en dos fases: en la primera se transfieren los ocho primeros bits a la zona izquierda del registro de direccionamiento, y en la segunda los ocho restantes a la parte derecha del citado registro.

¿Se puede transferir la información contenida en el acumulador directamente a memoria?

Generalmente, no. Es necesario utilizar un registro intermedio (que se le suele llamar tampón), al que se enviará la información para que en una segunda transferencia pase de este registro a memoria.

Registros de los microprocesadores

Almacenamiento temporal de datos binarios



En su acepción circuital, un registro no es más que un dispositivo para el almacenamiento temporal de información binaria. Su papel dentro del microprocesador es fundamental, puesto que es precisamente la dotación de registros internos la encargada de retener e incluso manipular la información en tratamiento por el cerebro integrado que denominamos microprocesador.

A través del presente capítulo se estudiará la naturaleza y especialización de los tipos de registros habitualmente utilizados en un microprocesador.

Registros de almacenamiento simple

Estos registros están diseñados únicamente para almacenar de forma temporal la información; por tanto, son los que tienen una estructura más sencilla. Sus principales características son las siguientes:

- Están conectados a bus, del que les llega la información o a través del que envían su contenido.
- Su funcionamiento se controla mediante una señal que autoriza la entrada de la información.
- Para descargar su contenido están conectados a un bus de salida. En algunos casos este bus coincide con el utilizado para la entrada de información.
- Para controlar la salida de información disponen, por ejemplo, de un conjunto de puertas lógicas AND a las que llega una señal que autoriza la descarga.

Normalmente funcionan en modo síncrono, y la entrada de información es controlada por una señal de reloj que permite la conexión del registro con el bus de entrada.

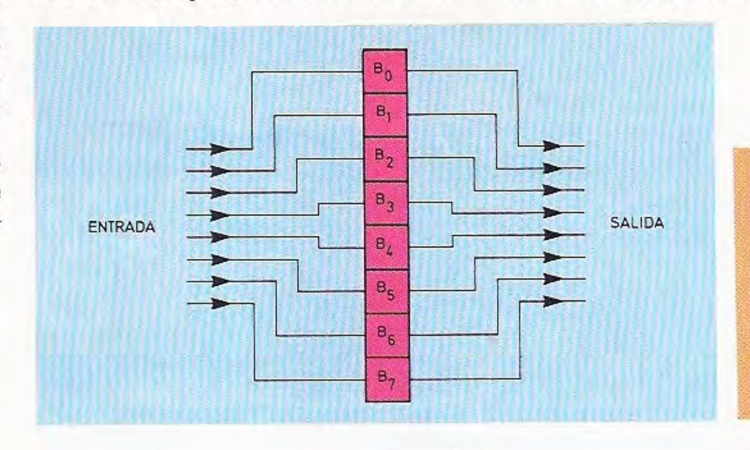
La capacidad del registro viene delimitada por el número de biestables de que disponga. En general, un registro de n biestables es capaz de almacenar 2^n configuraciones binarias distintas.

Registros de desplazamiento

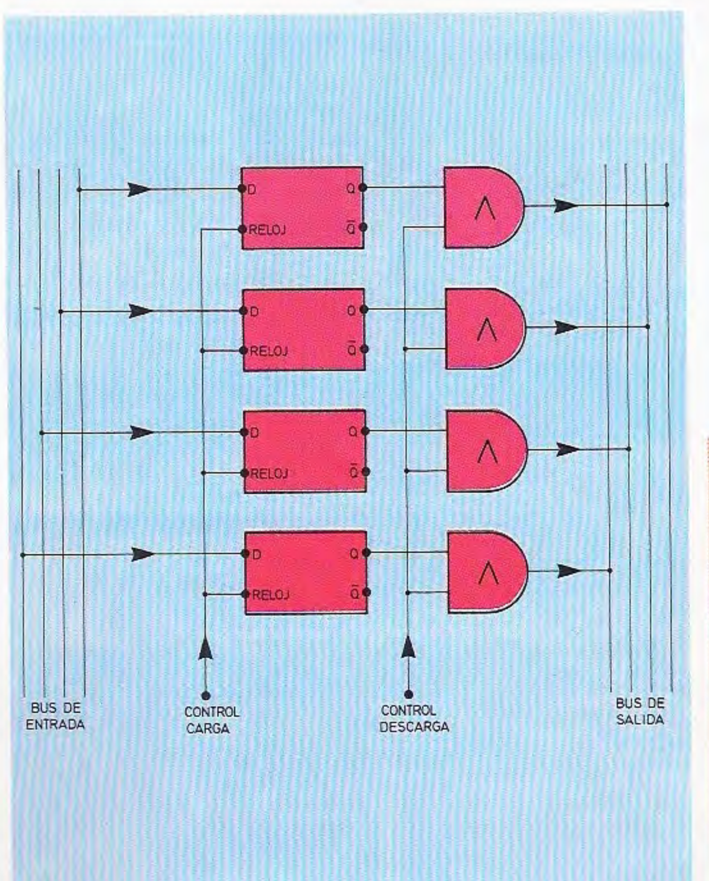
El funcionamiento de un registro de desplazamiento depende habitualmente de cuatro variables que definen la forma en que se desplazan los bits de dicho registro:

La primera variable puede tener dos valores: L (Lógico) o A (Aritmético). En el primer caso el desplazamiento afecta no sólo a los bits del registro, sino que también implica al bit de signo almacenado en un biestable exterior. En el segundo caso tan sólo se desplazan los bits del registro.

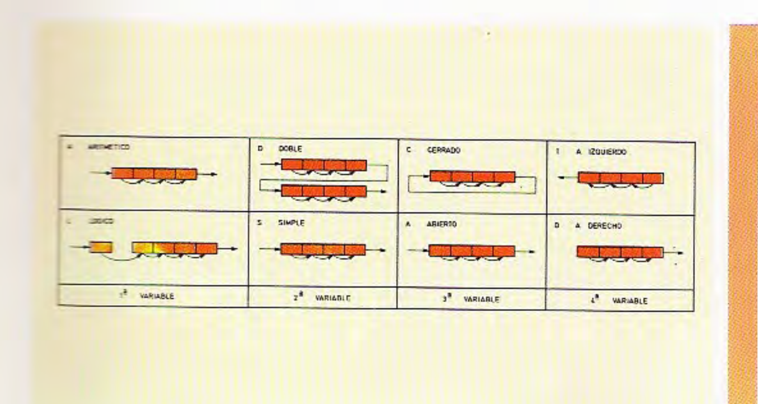
La segunda variable indica el número de registros que intervienen en el desplazamiento. Si su valor es S (Simple) sólo se desplaza un registro, pero si su



Esquema general de un registro para palabras binarias de ocho bits. Cada uno de los bits, cero o uno se almacena en un elemento biestable.



Registro de almacenamiento simple para palabras de cuatro bits. La línea de control autoriza la entrada de datos en el registro cuando toma el valor lógico uno. La salida de datos se realiza siempre que la señal de control de descarga tome el valor lógico uno.



Desplazamientos posibles en un registro de cuatro bits. Cada biestable, representado por un rectángulo, almacena un bit. Las flechas señalan la dirección de desplazamiento de los bits en el registro.

SITUACION INICIAL 1 0 0 1

DESPUES DEL 1ER DESPLAZAMIENTO 1 0 0 1 0

DESPUES DEL 2º DESPLAZAMIENTO 1 0 0 1 0 0

DESPUES DEL 3ER DESPLAZAMIENTO 1 0 0 1 0 0 0

DESPUES DEL 4º DESPLAZAMIENTO 1 0 0 1 0 0 0 0

Ejemplo de cuatro desplazamientos consecutivos de tipo ASAI (aritmético, simple, abierto, a izquierda), realizados en un registro de cuatro bits. El bit entrante en un desplazamiento aritmético toma siempre el valor cero.

valor es D (Doble) son desplazados los bits de dos registros.

La tercera variable controla la posible entrada de información para el desplazamiento, o la realización autónoma del mismo. Si esta variable toma el valor A (Abierto), la información contenida en el registro es «empujada» por un bit procedente del exterior (el último bit del registro «sale fuera»). En cambio, si toma el valor C (Cerrado), el último bit es el que inicia el desplazamiento colocándose en primer lugar (no sale información).

La cuarta variable define el sentido del desplazamiento, que puede ser D (Derecha) o I (Izquierda). En el primer caso los bits se desplazan una posición hacia la derecha y en el segundo, una posición hacia la izquierda.

Para definir completamente el desplazamiento de un registro se asigna a cada una de las cuatro variables uno de los dos posibles valores. De esta forma podemos llegar a definir hasta 16 combinaciones distintas:

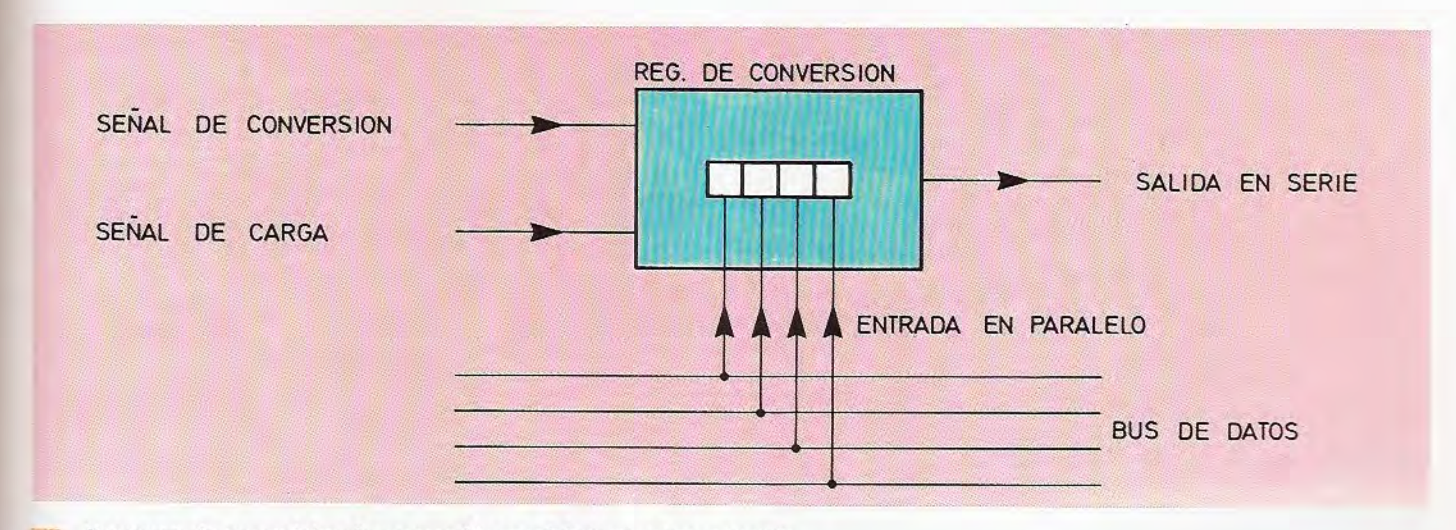
LSAD	LDAD	ASAD	ADAD
LSAI	LDAI	ASAI	ADAI
LSCD	LDCD	ASCD	ADCD
LSCI	LDCI	ASCI	ADCI

Registros de conversión paralelo/serie

La transmisión en paralelo consiste en el envío de todos los bits que componen la palabra original, simultáneamente, a las correspondientes posiciones del registro de destino. La transmisión en serie se realiza mediante una única línea de comunicación entre la palabra origen y la palabra destino, y se envían por ella, secuencialmente, cada uno de los bits.

Generalmente, los buses de los microprocesadores actúan en paralelo, mientras que determinadas unidades de entrada/salida transmiten los datos sobre una única línea en serie. Para enviar datos desde un registro del microprocesador hacia un registro de la unidad de entrada/salida es necesario realizar un cambio en el modo de transmisión: de paralelo a serie.

Para realizar esta conversión se siguen los siguientes pasos:



Esquema de un registro de conversión paralelo/serie de cuatro bits. Los datos entran en paralelo a través del bus de datos y salen, bit a bit, por la salida serie al activarse la señal de conversión.

- Inicialmente la información está colocada en un circuito paralelo (por ejemplo, en un bus).
- La CPU ordena la transmisión en paralelo del contenido del bus de datos a un registro de conversión situado en la unidad de E/S. Envía, para ello, una orden de carga.
- A continuación la CPU envía tantos impulsos como bits tenga el registro de conversión, con lo que los bits van saliendo del registro, uno a uno, por la línea de salida.
- Para realizar la conversión de una nueva palabra se coloca ésta en el bus de datos y se repite el proceso.

Registros de conversión serie/paralelo

La operación inversa a la anterior se efectúa en los registros de conversión serie/paralelo. Estos registros se utilizan normalmente para convertir la informa-

Resumen de los tipos de registros

Registros de almacenamiento simple

- Están conectados directamente al bus de datos.
- Mediante señales de carga/descarga pueden recibir información desde el bus, o transferir a éste los datos que tienen almacenados.
- Su funcionamiento es síncrono.

Registros de desplazamiento

- Se definen mediante cuatro variables:

1. Lógico o Aritmético.
2. Simple o Doble.
3. Abierto o Cerrado.
4. Derecha o Izquierda.

- El tipo de desplazamiento depende de los valores de estas cuatro variables. La misión encomendada a estos registros es el desplazamiento de la información contenida en sus bits, y en algún caso la admisión de un bit de entrada y/o un bit de salida.

Registros de conversión paralelo/serie

- Transforman un registro de n bits en n bits independientes que se pueden transmitir en serie.
- Su principal utilización es la conversión de la información enviada desde el microprocesador (paralelo) hacia un periférico (serie).
- Se utilizan normalmente en las unidades de entrada/salida.

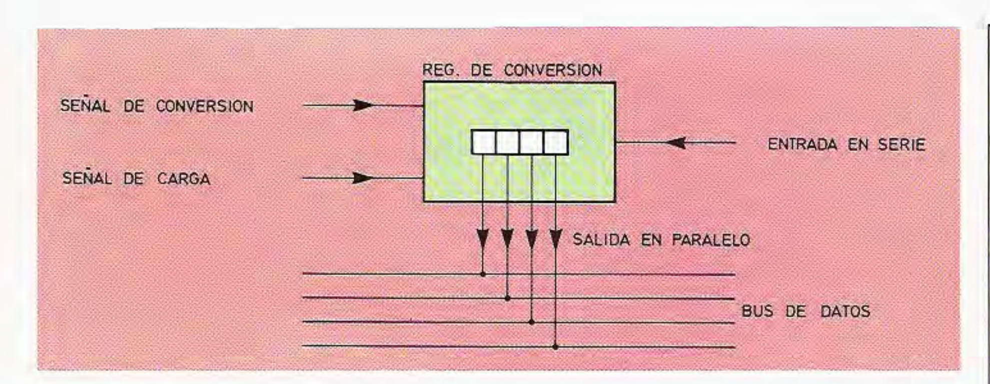
Registros de conversión serie/paralelo

- Almacenan un determinado número de bits recibidos en serie en un registro paralelo.
- Se suelen ubicar en las unidades de entrada/salida para recibir la información aportada por los periféricos (serie) y transformada en información procesable en el microprocesador (paralelo).

Registros contadores

- Su principal utilización son los registros contadores de instrucciones de la unidad de control del microprocesador.
- Pueden funcionar en dos modos:

— Modo «secuencial», que consiste en ir incrementando el valor que contienen en una unidad para que en todo momento apunte a la posición de memoria en que se encuentra la próxima instrucción.
— Modo «salto», que implica un cambio radical de su contenido como consecuencia de la ejecución de una instrucción de bifurcación.





Registro de conversión serie/paralelo. Los bits entran en serie a través de la única línea de entrada. Una vez que los cuatro bits están almacenados en el registro, salen hacia el bus de datos en formato paralelo.

ción en serie recibida de un periférico a información en paralelo operable por el microprocesador. El trabajo encomendado a este registro consiste en tomar un conjunto de bits que van llegando secuencialmente por una única línea de entrada, almacenarlos en su interior y mandarlos al bus correspondiente al completar la palabra.

La conversión serie/paralelo de datos se ejecuta de la siguiente forma:

- Se tiene una palabra de información recibida en serie y se desea transmitirla en paralelo.
- En primer lugar se almacena la palabra en el registro de conversión situado en la unidad de E/S. Para ello se envían tantos pulsos de reloj como bits tenga el registro, ya que la carga se realiza en serie.
- Para pasar la información desde el registro hasta el bus apropiado se dispone de un enlace que efectúa la transmisión en paralelo. Para realizar esta operación se utiliza también una línea de control.

Registros contadores

Un contador ordinal es cualquier dispositivo que genere una secuencia de configuraciones lógicas al ser activado por una señal. Análogamente, un registro contador es aquel capaz de generar una secuencia progresiva de configuraciones lógicas.

El registro contador más relevante es el registro contador de programas de la unidad de control del microprocesador, el cual almacena la posición de memoria en la que se encuentra la próxima instrucción a ejecutar. Este registro puede trabajar de dos formas: en modo «secuencial» y en modo «salto».

- *Modo «secuencial»*

En el registro se almacena inicialmente el valor de la dirección de memoria en la que se encuentra la primera instrucción del programa. Llamando n a esa posición de memoria y suponiendo que todas las instrucciones ocupan una única palabra, después de haber ejecutado la instrucción en curso el registro se incrementará en una unidad, con lo que su contenido pasará a $n + 1$, valor que coincide con la posición de memoria en la que se encuentra la siguiente instrucción.

- *Modo «salto»*

Si la instrucción ejecutada es una bifurcación, la siguiente posición de memoria a acceder vendrá indicada por la propia instrucción.

De esta forma el contenido del registro contador, después de ejecutar la instrucción, no tiene ninguna relación con el valor que tenía anteriormente. Contendrá una nueva dirección con la que comenzará de nuevo a trabajar en modo «secuencial», hasta que se ejecute otra instrucción de salto.

Superordenadores

En la actualidad se está produciendo una invasión del mercado informático por los ordenadores personales. Estos pequeños equipos son capaces de realizar varios cientos de operaciones por segundo. A la vez, están apareciendo otros ordenadores con una potencia increíblemente grande: los superordenadores, que pueden llegar a rebasar los 100 millones de operaciones por segundo. Estos grandes equipos son utilizados en centros especiales y tan sólo existen en la actualidad medio centenar de superordenadores funcionando en el mundo. Los dos más destacados son el CRAY-1, desarrollado por la CRAY RESEARCH INC., y el CYBER-205, construido por la CONTROL DATA CORPORATION.

La aparición de estos superordenadores ha permitido abordar problemas irresolubles anteriormente. Los campos más beneficiados han sido: la aerodinámica, la meteorología y la física atómica.

También ha sido posible realizar simulaciones de modelos matemáticos que representan un fenómeno natural. La NASA, por ejemplo, ha desarrollado, en el AMES RESEARCH CENTER, un modelo de flujo que muestra el comportamiento del aire cercano a la superficie de la parte posterior de un cohete. Para ello ha sido necesario trazar el flujo de aire alrededor de una red de 250.000 puntos, lo que supuso dieciocho horas de trabajo del superordenador ILLIAC IV, que llegó a realizar 10^{11} operaciones aritméticas.

A pesar de la enorme potencia de cálculo del ILLIAC IV, éste ha sido sustituido por el CRAY-1. El CRAY-1 está compuesto por un «bastidor principal» que contiene la unidad central de proceso y una estructura menor que aloja al sistema para la entrada y salida de información; este último compuesto, a su vez, por tres pequeños pero rápidos ordenadores y una extensión de memoria.

El único competidor del CRAY-1 es el CYBER-205. El precio en ambos casos oscila entre 10 y 15 millones de dólares. El servicio meteorológico de Gran Bretaña dispone de un CYBER-205 para el pronóstico del tiempo. Los usuarios de los superordenadores necesitan potencias de cálculo cada vez mayores. Es probable que en los próximos años lleguen a surgir nuevos equipos que conviertan a los «monstruos» CRAY-1 y CYBER-205 en pequeños ordenadores.

Métodos de direccionamiento

Buscando los datos en memoria

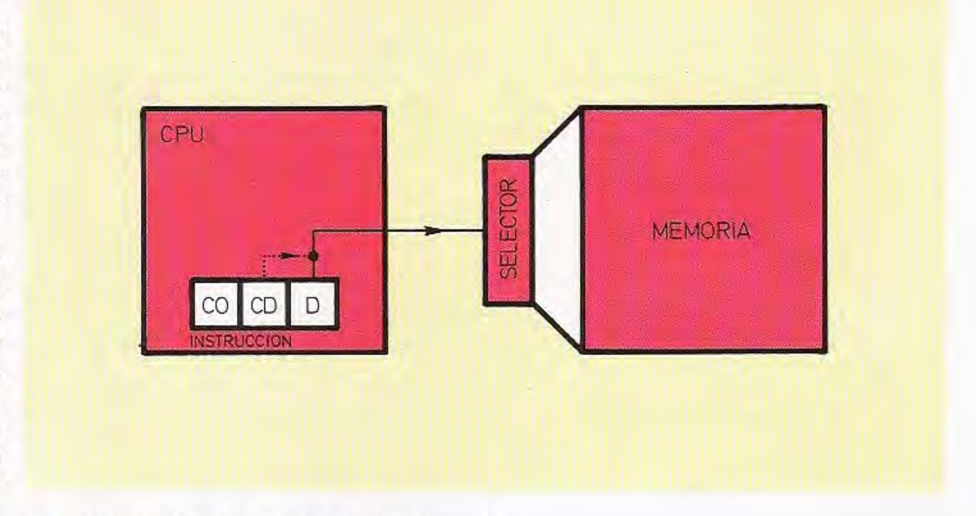


n aspecto esencial en la actividad que desarrolla el microprocesador lo constituye su relación con la memoria principal, de la que extrae o en la que deposita la instrucciones y datos.

El modo de determinar su emplazamiento y acceder a la información oportuna en cada instante es una cuestión de suma trascendencia y que se apoya en los denominados modos o métodos de direccionamiento del microprocesador. Estos afectan a la forma en la que éste se determinará, a partir de la instrucción en curso, la dirección de memoria afectada por la operación a ejecutar.

En general, se denomina método o modo de direccionamiento al procedimiento que se sigue para localizar el dato u operando a tratar a partir de la dirección apuntada por la instrucción de nivel máquina en curso.

Para seguir las descripciones que siguen conviene precisar que llamamos «dirección efectiva» (D) a la dirección de memoria en la que se encuentra el dato real a operar. Es conveniente recordar, asimismo, que el tratamiento que se dará a este dato lo determina el código de operación de la instrucción en curso (CO); en ocasiones, la naturaleza de la operación también es condicionada por otra zona de la instrucción denominada «condiciones de direccionamiento» (CD).



Direccionamiento implícito. La dirección efectiva se especifica directamente en la instrucción.

Principales métodos de direccionamiento

• *Direccionamiento implícito*

Este método de direccionamiento resulta el más sencillo, ya que la dirección efectiva es la especificada directamente en la zona de operando de la instrucción.

Una instrucción con este tipo de direccionamiento precisa un ciclo de acceso a memoria para leer el operando.

• *Direccionamiento inmediato*

Este método no es propiamente un direccionamiento. En la zona de dirección de la instrucción se encuentra almacenado el valor del operando; por tanto, no es necesario «buscar» en memoria el dato a procesar.

• *Direccionamiento indirecto*

En este caso la zona de operando de la instrucción no contiene la dirección de la posición de memoria en la que se encuentra el dato, sino la dirección de la posición de memoria en la que está almacenada la dirección efectiva.

La localización de un dato empleando direccionamiento indirecto requiere dos ciclos de memoria: uno para encontrar la dirección efectiva y otro para encontrar el dato a operar.

Cierto tipo de ordenadores permite el direccionamiento en cascada, es decir: la instrucción contiene la dirección de memoria en la que se encuentra una nueva dirección de memoria que direcciona a otra, ésta a otra, y así sucesivamente hasta llegar al último nivel que permite obtener la dirección efectiva. En este caso, el número de ciclos necesarios es función del número de niveles de la cascada.

• *Direccionamiento relativo*

Este método emplea dos direcciones distintas para calcular la dirección efectiva. A estas dos direcciones se las denomina dirección de referencia y dirección relativa. La primera indica el punto de memoria que debe ser considerado como dirección cero, y la segunda representa la dirección de memoria en la que se encontraría la información en el supuesto de que la dirección de referencia fuera la posición inicial de la memoria. De esta forma la dirección efectiva se obtiene sumando la dirección de referencia a la dirección relativa.

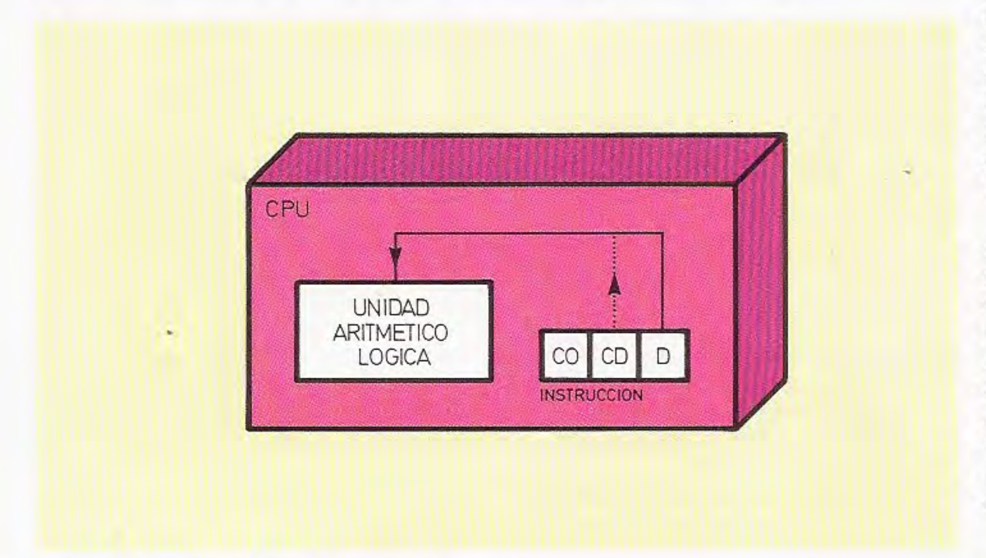
La técnica de direccionamiento relativo es muy útil cuando la longitud de palabra es insuficiente para direccionar toda la memoria. Por ejemplo, un ordenador con 16 bits de longitud de palabra y con un formato de instrucción que utilice 8 bits para el código de operación y condiciones de direccionamiento y otros 8 bits para la dirección, sólo puede direccionar $2^8 = 256$ posiciones de memoria. Lo normal en un ordenador de estas características es una capacidad de memoria de 65.536 palabras (64 K). Si no se puede pasar a un formato de instrucción de dos palabras, el direccionamiento relativo permite acceder directamente a cualquier posición de la memoria.

Veamos tres ejemplos de direccionamiento relativo:

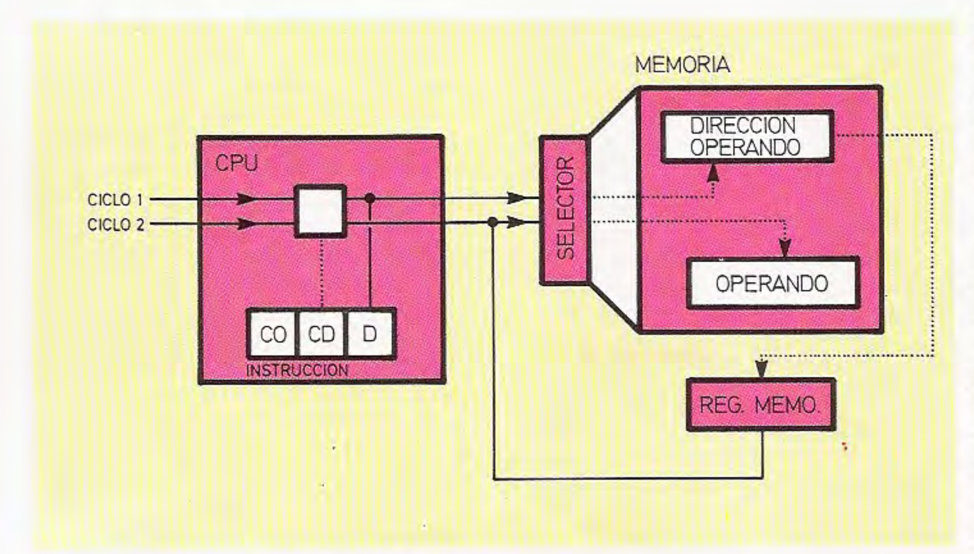
— Direccionamiento por base y desplazamiento. En este caso el ordenador

dispone de un registro, llamado registro de base, con la dirección de referencia (primera dirección de un programa o de una zona de datos, por ejemplo). La dirección contenida en la instrucción se llama, en este caso, desplazamiento, y la dirección efectiva se calcula sumando el contenido del registro de base al desplazamiento.

Algunos ordenadores disponen de varios registros de base y, por tanto, la instrucción debe indicar cuál de los registros de base es el utilizado para calcular la dirección efectiva.



En el direccionamiento inmediato no se hace referencia a ninguna posición de la memoria. La instrucción proporciona el dato de la operación en curso.



Direccionamiento indirecto. La zona de operando de la instrucción contiene la dirección de memoria en la que se encuentra almacenada la dirección efectiva.

— Direccionamiento por referencia al programa. La dirección de referencia, en este modo de direccionamiento, se encuentra almacenada en el contador de instrucciones.

En este caso se suelen utilizar dos zonas de direccionamiento. Una está situada después de la instrucción en curso, y la dirección efectiva se obtiene sumando al contenido del contador el desplazamiento. La segunda zona está situada delante de la instrucción en curso, y la dirección efectiva se calcula restando el desplazamiento al contenido del contador.

— Direccionamiento por página. En este caso se considera la memoria dividida en zonas de 2^n palabras, cada una de las cuales se denomina página. Normalmente, la zona de dirección de la instrucción es de n bits, de forma que puede direccionar a cualquiera de las posiciones de una página. La dirección efectiva se obtiene por yuxtaposición del número de página y de la zona de dirección de la instrucción.

- *Direccionamiento indexado*

En el direccionamiento indexado, la dirección efectiva se obtiene sumando al contenido de la zona de direccionamiento de la instrucción el contenido de un registro de la CPU llamado índice. Este registro índice admite varias operaciones directas, como carga, descarga, incremento en una unidad, decremento en una unidad, etc.

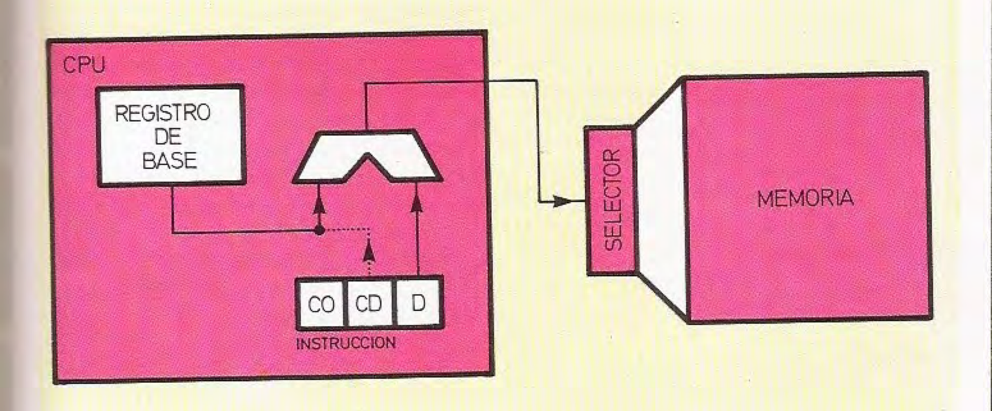
La principal utilidad de esta técnica estriba en que los programas pueden tratar, mediante una única instrucción contenida en un bucle, datos consecutivos almacenados en la memoria principal. La dirección especificada en la instrucción es la de la primera posición de memoria, y a ella se le suma el contenido del registro índice (a este tipo de proceso se le denomina, precisamente, indexación), que inicialmente vale cero y que se incrementa en una unidad cada vez que se ejecuta un paso completo del bucle. La última instrucción del bucle se encarga de comprobar si el índice contador de pasos es superior al número de elementos almacenados consecutivamente en la memoria; en caso negativo inicia un nuevo paso, y en caso afirmativo da por finalizada la ejecución del bucle.

Mediante la utilización de varios registros índices se pueden ejecutar bucles anidados.

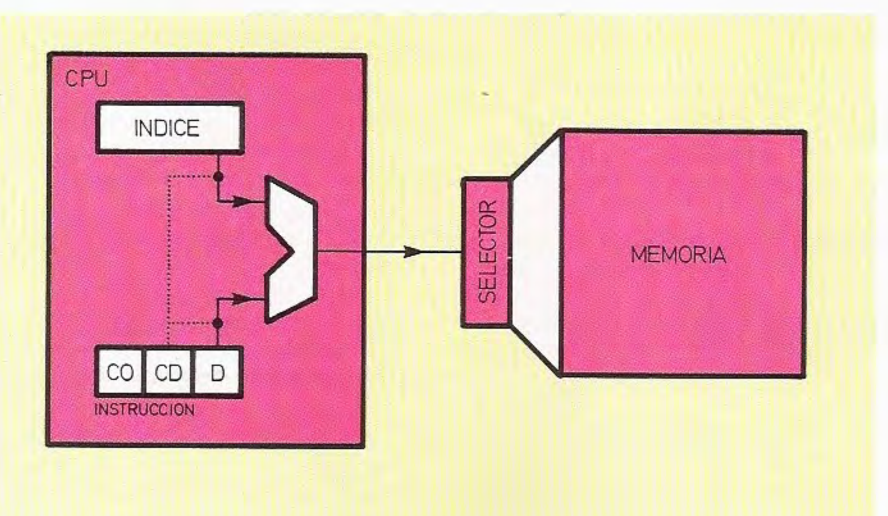
Con objeto de economizar costes, en algunos microordenadores los índices se implantan en la memoria principal.

Combinaciones de los diferentes métodos de direccionamiento

Los métodos de direccionamiento descritos anteriormente, aunque sí son los más empleados, no son los únicos.



Dirección por base y desplazamiento. Este tipo de recursos tiene especial utilidad en multiprogramación. Permite reubicar los programas dentro de la memoria principal.



En el direccionamiento indexado la posición efectiva se obtiene sumando el contenido de la zona de direcciones de la instrucción al contenido de un registro índice.

Teoría de la complejidad

El objeto fundamental de la teoría de la complejidad es establecer una clasificación entre los problemas según su dificultad, de forma que se puede determinar cuándo un problema es intratable. Para concluir esto, debe demostrarse primero que es equivalente, en complejidad, a otro problema que pertenezca a la clase de los intratables.

Aunque saber que un problema es intratable no ayuda a resolverlo, sí permite replantear los objetivos y dosificar los esfuerzos necesarios para resolverlo.

Para enunciar un problema de un modo formal se deben especificar dos grupos de datos:

Entrada: Descripción de todos los parámetros necesarios para plantear el problema.

Cuestión: Planteamiento de los objetivos del problema en función de los parámetros de la entrada.

Por ejemplo, un célebre problema, el del viajante de comercio (VC), se puede enunciar de la siguiente manera:

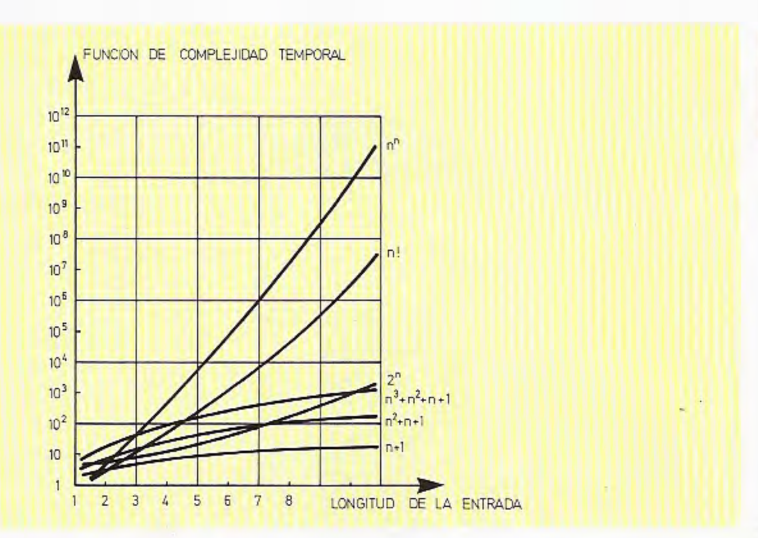
Entrada: Sea $C = \{c_1, \ldots, c_m\}$ un conjunto de ciudades, y sea d_{ij} la distancia existente entre cualquier par de ciudades c_i y c_j.

Cuestión: Encontrar una ordenación de las ciudades $< c_{k1}, c_{k2}, \ldots, c_{km} >$ tal que minimice la siguiente expresión:

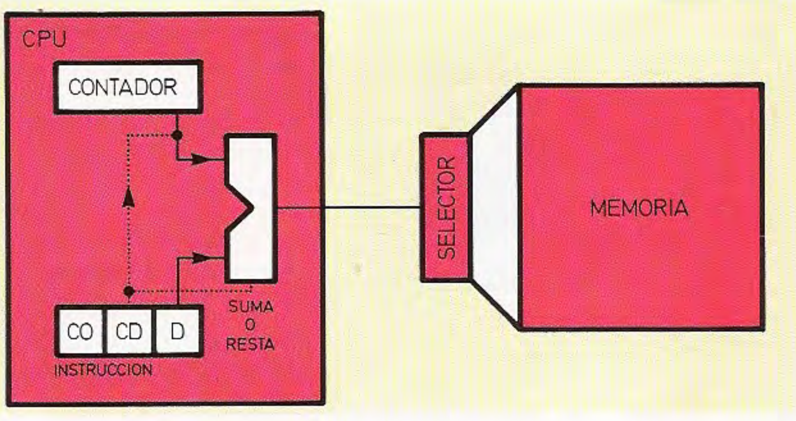
$$\left[\sum_{i=1}^{m-1} d_{k_1, k_{i+1}} \right] + d_{k_m, k_1}$$

El problema del viajante de comercio se reduce a calcular el orden en el que el viajante debe recorrer todas las ciudades para que, pasando una única vez por cada una de ellas, el recorrido sea mínimo y finalice en la misma ciudad de la que partió.

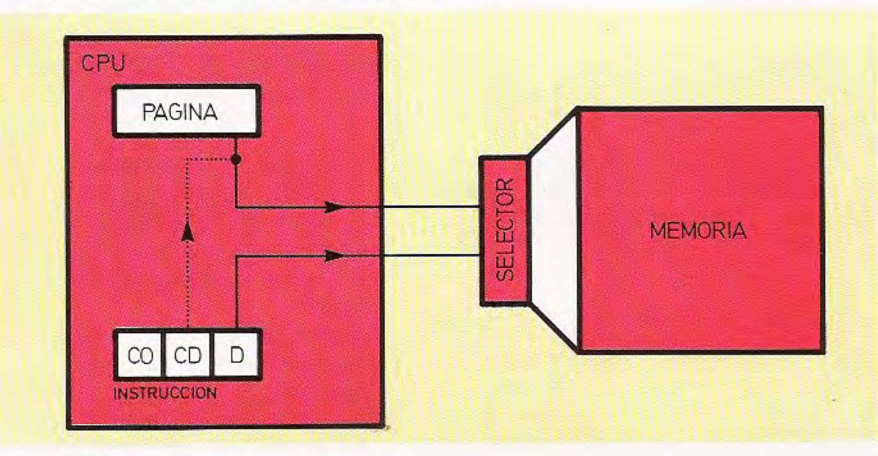
El método utilizado por la teoría de la complejidad para clasificar un problema consiste en asociarle un lenguaje formal. Las matemáticas ofrecen en este campo el soporte teórico adecuado para la asociación de un lenguaje de forma casi automática a los problemas decibles, es decir, aquellos cuya solución es «SI» o «NO».



La curva representa el tiempo de ejecución de un algoritmo en función de la longitud de entrada. El tiempo sirve, además, como parámetro de calidad del método.



El direccionamiento por referencia al programa es una modalidad del indirecto. El operando se almacena en la posición de memoria cuya dirección coincide con el contenido del contador de instrucciones.



Direccionamiento por página. Esta modalidad permite ampliar la memoria virtual de los ordenadores.

Cada microprocesador incorpora métodos particulares que normalmente se basan en combinaciones de los distintos tipos principales de direccionamiento.

El direccionamiento relativo y el direccionamiento indexado son iguales desde un punto de vista operativo, ya que en ambos casos la dirección efectiva se consigue sumando el contenido de la zona de direccionamiento de la instrucción al contenido de un registro (base o índice). Si bien, desde el punto de vista funcional, son completamente distintos, ya que el primero permite utilizar diferentes direcciones en función de una sola dirección de referencia, mientras que el segundo permite acceder sucesivamente a posiciones secuenciales de memoria, cuya dirección viene indicada por la propia instrucción.

Una de las combinaciones más usuales de los métodos de direccionamiento es el denominado indexado indirecto. Como su propio nombre indica, este método consiste en sumar al contenido del registro índice el contenido de la zona de direccionamiento de la instrucción. El valor así obtenido no es la dirección efectiva, sino la posición de memoria cuyo contenido es la dirección efectiva.

Otra combinación muy común es la de los métodos relativo e indexado, en la que intervienen la dirección de referencia, la dirección relativa y el contenido del registro índice. Mediante este tipo de combinaciones se obtiene un funcionamiento que participa de las dos características funcionales descritas anteriormente para los direccionamientos relativo e indexado.

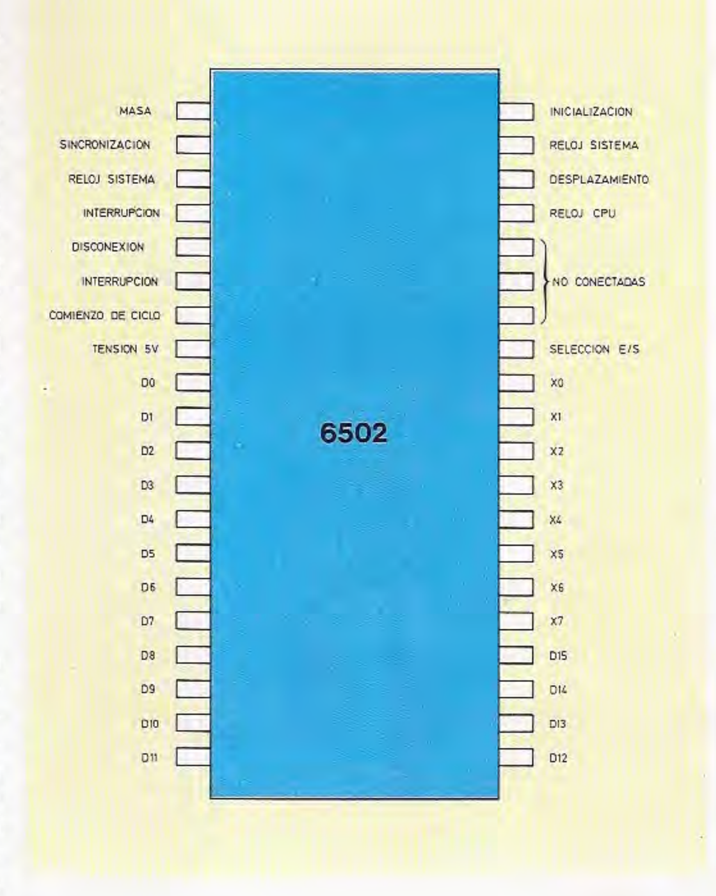
Microprocesadores de 8 y 16 bits

En los cimientos del ordenador personal



Al estudiar la arquitectura de los sistemas microordenadores se observó que su CPU estaba constituida por un microprocesador. Son muchos y muy diversos los microordenadores y ordenadores personales que han visto la luz en los últimos años. Y también son variados los microprocesadores que la industria microelectrónica ha desarrollado para cumplir el papel de cerebro integrado. No obstante, es curioso constatar que son un breve puñado tan sólo los microprocesadores que alcanzan un verdadero protagonismo en el universo microinformático. Valga como botón de muestra el hecho de que más de la mitad de los microordenadores actualmente en el mercado utilizan el microprocesador de 8 bits Z-80 o el 8088 de 16 bits.

En los próximos párrafos se concreta el estudio teórico realizado sobre el microprocesador en tres sencillos modelos, ya tradicionales aunque perfectamente aptos para ilustrar la naturaleza real de este cerebro integrado. Una vez terminado el recorrido por el 6502, Z-80 y 6809, se esbozarán las especificaciones básicas extensivas a la mayor parte de los microprocesadores de 16 bits.



Esquema de las conexiones externas del microprocesador 6502. El bus de direcciones se ha representado con la letra D (D0, D1, ..., D5) y el de datos con la letra X (X0, ..., X7).

Microprocesador 6502

- *Generalidades*

La organización interna del microprocesador 6502 es muy parecida a los de la familia 6800 de Motorola, si bien a nivel de software estos dos microprocesadores no son totalmente compatibles.

La tecnología de fabricación del 6502 es N.MOS y se alimenta con una única fuente de 5 V. La pastilla en que está instalado dispone de 40 conexiones con el exterior, a través de las que el microprocesador recibe y suministra la información. La capacidad de direccionamiento del 6502 es de 64 Kbytes.

- *Registros*

El microprocesador 6502 dispone de cinco registros internos de 8 bits y uno de 16 bits, distribuidos de la siguiente forma:

- 2 registros índice (de 8 bits).
- 1 puntero de pila (de 8 bits).
- 1 acumulador (de 8 bits).
- 1 contador de instrucciones (de 16 bits).
- 1 registro de estado o de código de condiciones (de 8 bits).

El puntero de pila contiene la última posición de una zona de la memoria RAM gestionada a modo de pila.

El acumulador es utilizado por el microprocesador para efectuar las operaciones aritmético-lógicas.

El contador de instrucciones se encarga de «apuntar» a la posición de memoria que contiene el código de operación de la próxima instrucción a ejecutar. Al disponer de 16 bits, el máximo número que puede llegar a almacenar es 2^{16} = 65.536, que corresponde al espacio total direccionable por el microprocesador 6502 (65.536/1.024 = 64 Kbytes).

El último registro interno es el destinado a almacenar la información codificada del estado del microprocesador. Cada uno de sus 8 bits tiene un significado distinto y, según contenga un 0 o un 1, indica que la condición que representa está desactivada o activada, respectivamente. Las distintas condiciones son las siguientes:

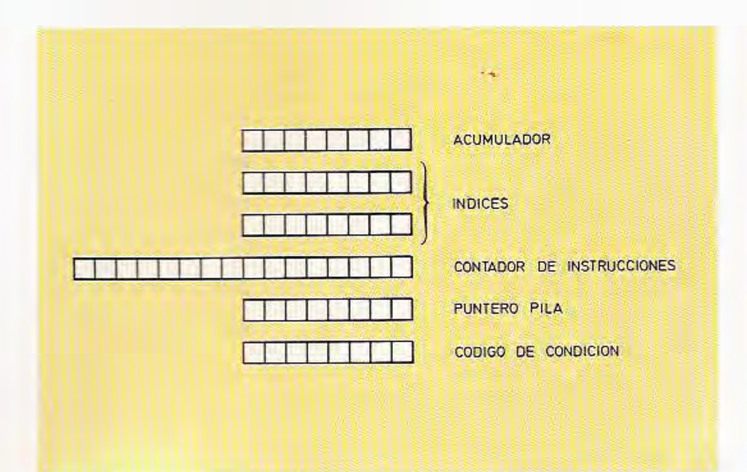
— Acarreo (C).

— Cero (Z). Si está a uno indica que el resultado de la última operación ha sido nulo.

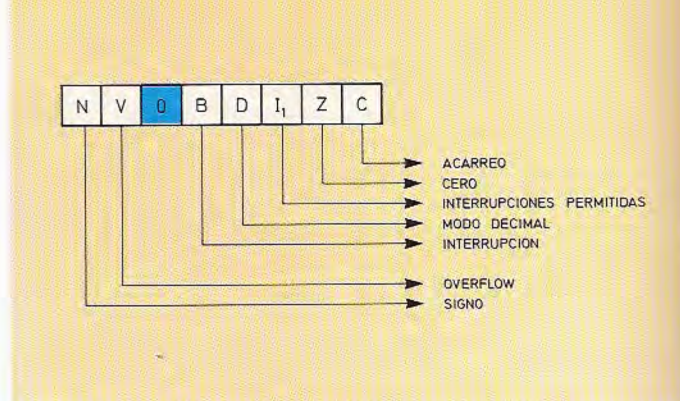
— Interrupción (I). Permite o inhibe las posibles interrupciones durante la ejecución del programa.

— Modo (D). Cuando este bit vale 0 las operaciones se realizan en modo binario; en cambio, cuando vale 1 el modo es decimal.

— Interrupción (B). Si vale 1 el programa se interrumpe.



Organización de los registros internos del microprocesador de 8 bits 6502.



Registro de estado del microprocesador 6502. Cada uno de los elementos biestables almacena un bit de condición cuyo significado se indica en la figura.

— Desbordamiento u «overflow» (V). Indica que el resultado de la operación efectuada excede la capacidad máxima del microprocesador.

— Signo (N). Sirve para controlar el signo (positivo o negativo) del resultado de una operación.

- *Posibilidades de direccionamiento*

El microprocesador 6502 admite ocho tipos distintos de direccionamiento:

— Implícito.
— Inmediato.
— Extendido.
— Indirecto.
— Directo.
— Indexado.
— Indexado indirecto.
— Relativo.

- *Juego de instrucciones*

El juego básico de instrucciones del 6502 está constituido por 56 instrucciones distintas.

En cuanto a la posibilidad de interrupciones, el 6502 admite dos tipos distintos: enmascarables (por software) y no enmascarables (por hardware).

Microprocesador Z-80

Este microprocesador fue comercializado en 1976 por la firma ZILOG. El Z-80 integra 8.000 transistores y es básicamente una ampliación del 8080 de Intel. Su tamaño es aproximadamente un 20 por 100 más grande que el 8080. Admite todas las instrucciones del clásico 8080 y algunas más.

Está fabricado con tecnología N.MOS y se alimenta con 5 voltios. Está integrado en una pastilla con 40 conexiones al exterior y puede gestionar una memoria de hasta 64 Kbytes.

Una característica esencial de este microprocesador es que dispone de varios registros banalizados, de uso general, con los que consigue una gran versatilidad.

- *Registros*

El Z-80 posee un total de 22 registros internos, parte de los cuales están duplicados. Dentro de los registros duplicados cabe destacar dos juegos de seis registros de utilización general, cada uno de ellos de 8 bits, que pueden utilizarse concatenados formando registros dobles de 16 bits.

Los 22 registros del Z-80 son los siguientes:

— 2 registros índices I_1 e I_2 (de 16 bits).
— 1 registro puntero de pila SP (de 16 bits).
— 1 registro de dirección de página PA (de 8 bits).
— 2 registros acumuladores A_1 y A_2 (de 8 bits).
— 2 registros de estado E_1 y E_2 (de 8 bits).
— 2 juegos de 6 registros generales (de 8 bits) G_1, G_2, ..., G_{12}.
— 1 registro contador de instrucciones PC (de 16 bits).
— 1 registro contador «de refresco» R (de 8 bits).

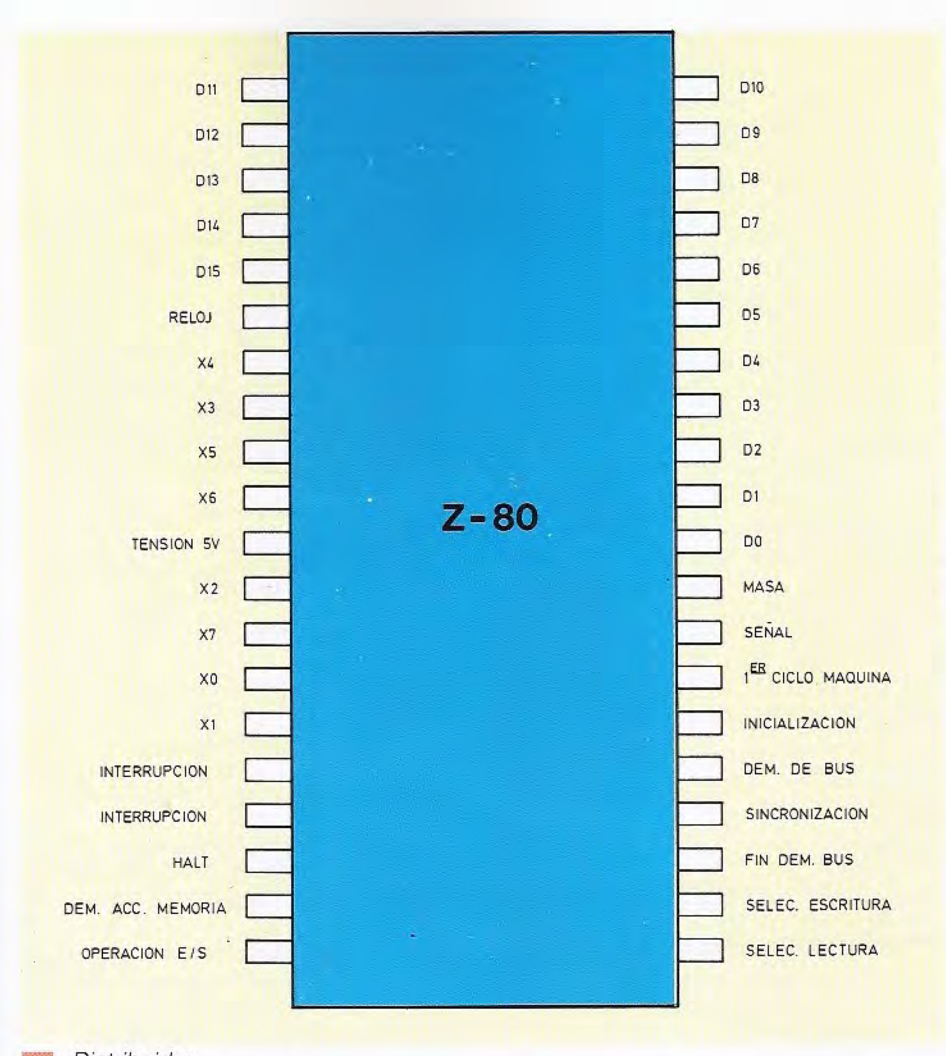
Los registros I_1 e I_2 se utilizan para facilitar algunos tipos de direccionamiento, y el puntero de pila P sirve para gestionar la pila.

Mediante el registro de direccionamiento de página PA se puede acceder a la primera posición de un bloque de memoria principal (página).

Los registros acumuladores y de código de estado están duplicados en el microprocesador Z-80. Por supuesto, la función encomendada a ambos registros es la misma que la de cualquier acumulador o registro de código de estado. El significado de los distintos bits de cada uno de estos dos últimos registros es el siguiente:

— Acarreo (C).
— Sustracción (N).
— Desbordamiento (V).
— Semiacarreo (H).
— Cero (Z).
— Signo (S).

El Z-80 posee además un registro contador de instrucciones de 16 bits y



Distribuidor de patillas del chip microprocesador Z-80.

un «contador de refresco» de 8 bits para memorias dinámicas.

- *Posibilidades de direccionamiento*

Seis son los modos de direccionamiento en los que se apoya el Z-80 para acceder a los 64 K que constituyen el espacio de memoria direccionable:

— Implícito.
— Inmediato.
— Directo.
— Por registros.
— Indexado.
— Relativo.

- *Juego de instrucciones*

El juego de instrucciones del Z-80 es más potente que el del microprocesador 6502. Dispone de 158 instrucciones básicas, que se convierten en 696 al combinarlas con las distintas posibilidades de direccionamiento.

Otra de las características fundamentales del Z-80 es su juego de tres interrupciones, para cuya activación dispone de dos conexiones en la pastilla.

Microprocesador 6809

En su nacimiento, el 6809 constituyó una verdadera progresión en el terreno de los microprocesadores; combinaba la disposición de sus buses externos tradicional en los chips de 8 bits (bus de datos de 8 bits y bus de direcciones de 16 bits) con una potente arquitectura interna de 16 bits.

La tecnología en que está realizado es H.MOS y se alimenta con una tensión de 5 V. La pastilla tiene 40 conexiones hacia el exterior.

Como integrante de la familia 6800, puede utilizar la mayoría de los componentes diseñados para otros microprocesadores «hermanos».

- *Registros*

Como ya se ha mencionado, casi todos los registros internos del 6809 tienen una longitud de 16 bits; su número se eleva a nueve y el cometido de cada uno de ellos es el que se apunta a continuación:

— 2 registros índices X e Y (de 16 bits).
— 2 punteros de pila U y S (de 16 bits).
— 2 acumuladores A y B (de 8 bits), que cuando se concatenan pueden utilizarse como un acumulador de 16 bits.
— 1 registro contador de instrucciones CI (de 16 bits).
— 1 registro de estado ICC que indica el código de condición (de 8 bits).
— 1 registro de «dirección de página» DP (de 8 bits).

Veamos el funcionamiento de cada uno de estos registros:

Los registros de 16 bits X e Y se utilizan principalmente para el direccionamiento indexado; no obstante, también sirven ocasionalmente para realizar algunas operaciones aritméticas y lógicas.

Los punteros de pila U y S, que también disponen de 16 bits, desempeñan la labor típica de este tipo de registros. El registro S sirve para la pila de órdenes del sistema (por ejemplo, interrupciones), mientras que el registro U es utilizado para controlar la pila del usuario.

Los acumuladores de 8 bits, A y B, se utilizan para realizar las operaciones aritméticas y lógicas. Presentan la posibilidad adicional de funcionar concatenados y, de esta forma, se convierten en un único acumulador de 16 bits. El re-

gistro contador de instrucciones CI también está formado por 16 bits y funciona como cualquier otro contador de instrucciones de un microprocesador: direcciona la posición de memoria en que se encuentra la siguiente instrucción a ejecutar.

El registro de dirección de página DP se utiliza para direccionar la página de memoria en que se encuentran los datos: apunta al primer octeto de un grupo de 256 octetos donde se encuentra la información buscada.

El registro CC agrupa a 8 bits; cada uno de ellos tiene un significado específico:

— Acarreo (C). Es el bit utilizado para el acarreo tanto de la suma como de la resta.

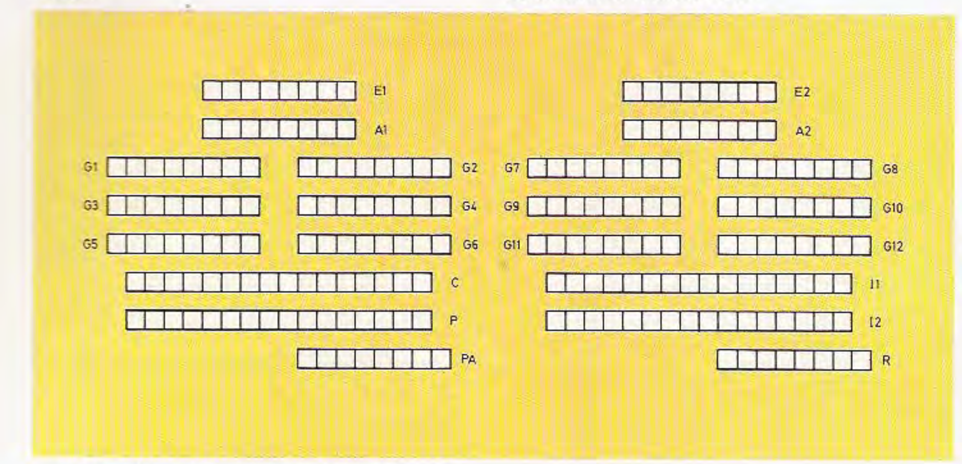
— Overflow (V). Este bit indica si el resultado de una operación excede de la magnitud máxima tratable. Cuando se produce un overflow pasa a valer 1; en caso contrario, vale 0.

— Cero (Z). Cuando el resultado de una operación es cero, pasa a valer 1; si el resultado es distinto de cero, pasa a valer 0.

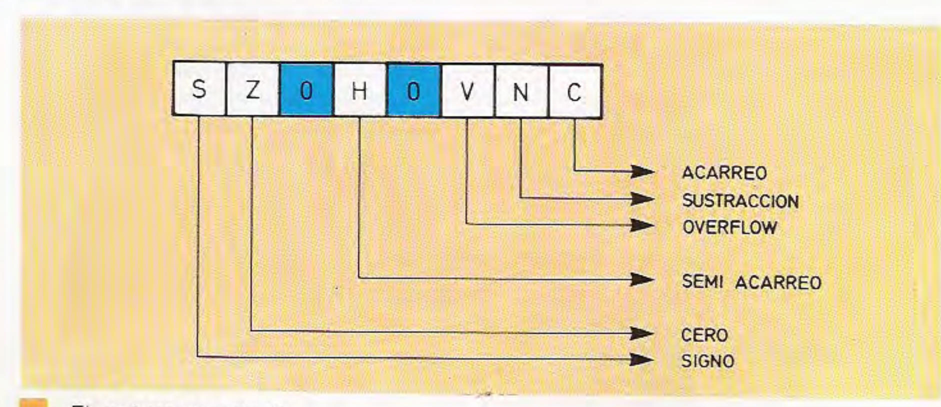
— Negativo (N). Indica el signo del resultado de la operación ejecutada (0 si es positivo y 1 si es negativo).

— Interrupción (I). Mediante su valor se gestionan los distintos tipos de interrupciones.

— Semiacarreo (H). Se utiliza para identificar el acarreo intermedio en determinado tipo de instrucciones que actúan sobre cuatro bits.



El doble juego de registros internos constituye una de las características básicas del microprocesador Z-80. A los registros generales (G1. .... G6) se les suele llamar B. C. D. E. H y L.



El registro de estado del microprocesador Z-80 consta de 8 bits de los que sólo se utilizan seis.

— Interrupción rápida (F). Se encarga de gestionar otro tipo de interrupciones distinto al de las controladas por el bit 1.

— Tipo de copia (E). Si su valor es 1 indica que en la interrupción se ha producido una copia de seguridad de todos los registros del 6809. Si vale 0 indica que tan sólo se han copiado los registros CI y CC.

• *Posibilidades de direccionamiento*

El microprocesador 6809 dispone de un total de diez modos de direccionamiento:

— Implícito.
— Inmediato.
— Extendido.
— Indirecto extendido.
— Directo.
— Por registros.
— Indexado.
— Indexado indirecto.
— Relativo.
— Relativo indexado.

• *Juego de instrucciones*

El juego de instrucciones del 6809 se puede descomponer en cinco grandes grupos:

— Operadores aritméticos y lógicos sobre palabras de 8 bits (memoria, acumuladores A y B, registro de estado CC y registro de dirección de página DP).

— Operadores aritméticos y lógicos que trabajan con un registro acumulador compuesto por la concatenación de A y B y, por tanto, de 16 bits.

— Operaciones relativas a los registros índices X e Y y a los registros punteros de pila U y S.

— Operaciones de bifurcación, tanto condicionales como incondicionales.

— Operaciones de sincronización e interrupción.

Microprocesadores de 16 bits

En el mundo de los microprocesadores de 16 bits cabe distinguir dos categorías. Una de ellas integrada por los microprocesadores nacidos como ampliación o prolongación de anteriores chips de 8 bits y que comparten un gran nú-

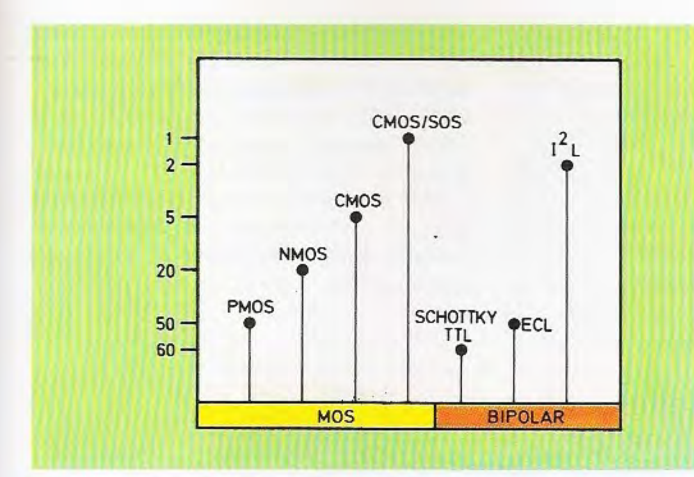
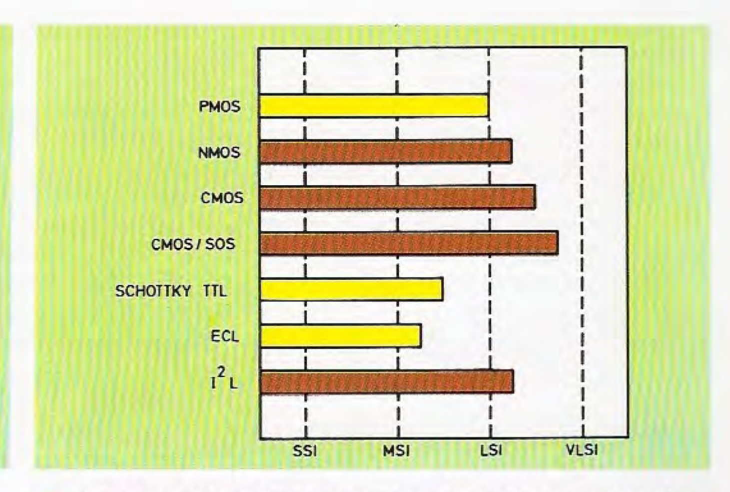


Gráfico representativo del factor de calidad de las principales familias tecnológicas. El factor de calidad relaciona la velocidad de la operación con el consumo de energía.



En el eje horizontal se representa el grado de integración comparativo de componentes electrónicos que alcanza cada una de las familias lógicas de las tecnologías bipolar y MOS, habitualmente utilizadas en la fabricación de microprocesadores.



Familia de circuitos integrados lógicos

Una familia tecnológica identifica a un conjunto de circuitos elaborados según una misma técnica.
El término tecnología se asocia a todo proceso de fabricación. Por tanto, los conceptos de tecnología y familia tecnológica son diferentes. Dentro de la tecnología bipolar, TTL, por ejemplo, existen diversas familias: TTL estándar, TTL Schottky, etc.

Tecnología bipolar

Los circuitos de esta tecnología emplean como elemento básico el transistor bipolar. Sus principales familias son las siguientes:

- Familia RTL. Esta familia proporcionó la base para la fabricación de los primeros circuitos integrados.
- Familia DTL. Se desarrolló simultáneamente a la familia RTL. El principal problema de la DTL es que permite una densidad de integración muy reducida.
- Familia TTL. Esta familia es la más difundida en la actualidad. La entrada al operador se efectúa a través de un transistor multiemisor.
- Familia TTL Schottky. Utiliza la misma lógica que la familia TTL estándar, incorporando además diodos Schottky con el fin de incrementar la velocidad de transición de un estado a otro.
- Familia ECL. La más rápida de todas las familias de la tecnología bipolar.
- Familia IIL. Utiliza pares de transistores acoplados sobre un mismo sustrato de silicio. La velocidad de esta familia no es demasiado elevada, pero con ella se consiguen elevadas densidades de integración.

Tecnología MOS

Esta tecnología es la más utilizada en la fabricación de microprocesadores. Sus famil as más importantes son las siguientes:

- Familia P.MOS, basada en transistores MOS de canal P. La velocidad de esta familia es inferior a la de la familia TTL-Schottky. Consume, sin embargo, menos potencia, su estructura es mucho más simple y su nivel de integración es muy alto.
- Familia N.MOS. Está basada en transistores MOS de canal N. Su estructura es muy similar a la de la familia P.MOS; pero la velocidad de transición, mucho más elevada.
- Familia C.MOS. Utiliza transistores P.MOS y N.MOS. Su consumo es mínimo y el nivel de integración es igual o incluso mayor que en la familia N.MOS. La velocidad de funcionamiento es sensiblemente inferior.
- SOS. Esta familia surge como un perfeccionamiento tecnológico de la familia C.MOS para aumentar su velocidad.

Tecnología CTD

Esta tecnología, más moderna que las anteriores, es el resultado de los últimos avances conseguidos en microelectrónica.
Su estructura es más simple que la de las restantes tecnologías y resulta, por tanto, más económica y alcanza un nivel más alto de integración.
Sus dos familias fundamentales son las siguientes:

- Familia CCD.
- Familia BBD. Es similar a la familia CCD. Se diferencia de ella en la disposición complementaria de las zonas de carga y en los electrodos para la polarización sucesiva.

mero de características con sus predecesores; aquí cabe considerar, por ejemplo, a los fabricados por la compañía americana Intel.

Otro grupo es el que engloba a los productos de 16 bits de fabricantes como Zilog y Motorola, cuyos diseños difieren netamente de los aplicados a los microprocesadores de 8 bits anteriormente comercializados por las referidas compañías.

croprocesador 68000 de la firma Motorola.

En el mundo de los equipos domésticos, el predominio se mantiene en el escenario de los 8 bits, con los microprocesadores Z-80 y 6502 a la cabeza.

Hace escasamente cinco años sólo existían prácticamente microprocesadores de 8 bits. La llegada de los chips de 16 y 32 bits plantearon la necesidad de desarrollar nuevos sistemas operativos, compiladores de lenguajes, programas de utilidad, etc., adaptados a las potencialidades de estos microprocesadores. Un proceso que supuso, en definitiva, la plena renovación de la realidad informática y que resulta indisociable de la evolución en esta disciplina.

A la hora de desarrollar un nuevo bagaje de software para un determinado modelo de microprocesador, se impone observar las siguientes consideraciones:

• Los microprocesadores se fabrican

Usuarios de microprocesadores de 16 bits

Existen dos tipos de sistemas de ordenador en los que hoy en día están presentes los microprocesadores de 16 bits:

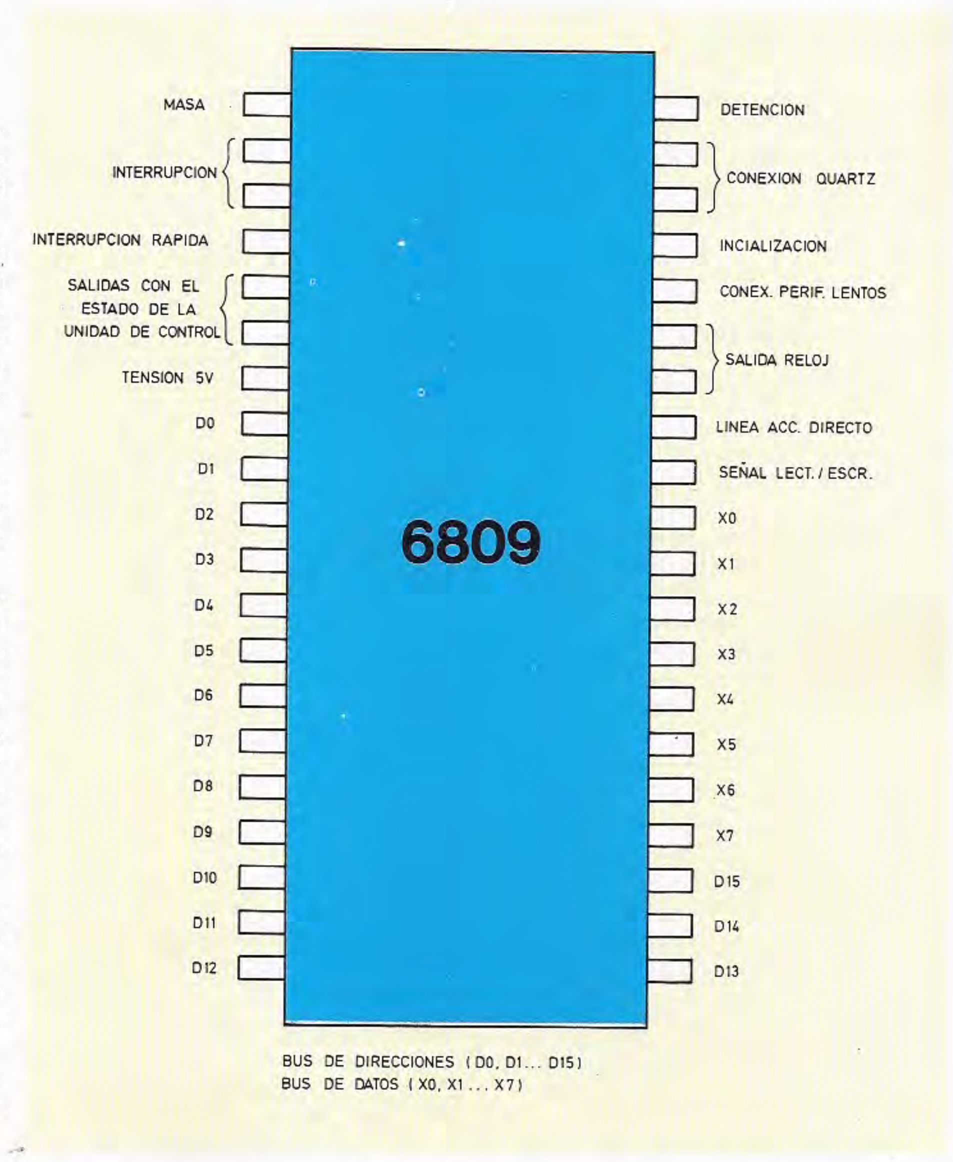
• Miniordenadores y ordenadores personales. Mediante la utilización de un microprocesador de 16 bits como CPU, estos sistemas consiguen mejorar la rapidez, aumentar la capacidad de direccionamiento y el número de instrucciones procesables.

Los equipos así diseñados son capaces de mecanizar toda la gestión de empresas pequeñas y medianas sin necesidad de recurrir a ordenadores más grandes y caros.

Los ordenadores personales basados en un microprocesador de 16 bits —un perfecto botón de muestra lo constituyen los ordenadores de la familia IBM-PC y sus compatibles— sirven también como potentes herramientas de trabajo para profesionales independientes, que han comprobado ya cómo el uso de estas máquinas sobrepasa a la simple informática doméstica, educativa o de entretenimiento.

• Otro marco de actuación de los microprocesadores de 16 bits son los grandes ordenadores, que consiguen, gracias a ellos, descargar de trabajo a la CPU e incrementar la productividad del sistema.

En la actualidad, el mercado de los ordenadores personales de uso profesional o de gestión está firmemente dominado por los microprocesadores de 16 bits (8088, 8086, 80286...), si bien cada vez es más acusada la presencia de chips de 32 bits, particularmente el mi-



Esquema de conexiones externas del microprocesador 6809. El bus de direcciones se ha representado con la letra D (D0, D1, ..., D15), y el de datos con la letra X (X0, X1, ..., X7).

de acuerdo a un juego de instrucciones internas determinado, y no al revés.

- No hay que pasar por alto la capacidad de integración de los nuevos componentes en equipos antiguos, y viceversa.

Los 16 bits en escena

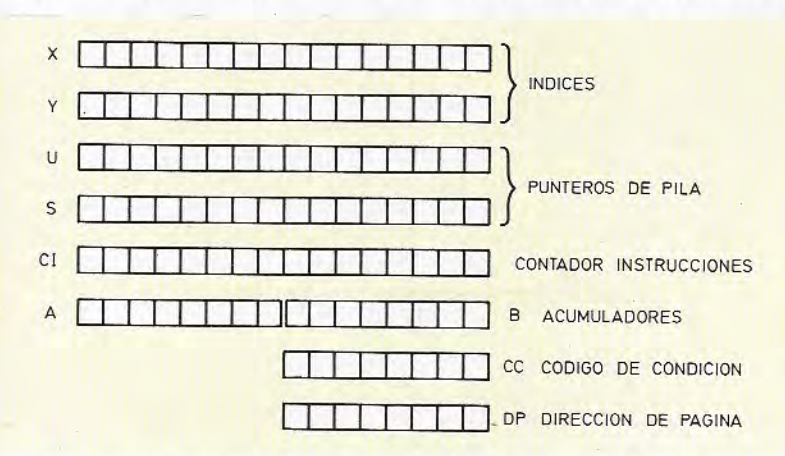
Los microprocesadores de 8 bits tienen normalmente una capacidad de direccionamiento de 64 Kbytes. Esta capacidad se ha visto incrementada al llegar los micros de 16 y 32 bits: el INTEL 8086 puede direccionar hasta 1 Mbyte; el MOTOROLA 68000, hasta 16 Mbytes, y el ZILOG Z-8000, hasta 48 Mbytes.

En cuanto a la tecnología empleada en la fabricación de los microprocesadores de 16 bits, cabe distinguir dos categorías esenciales:

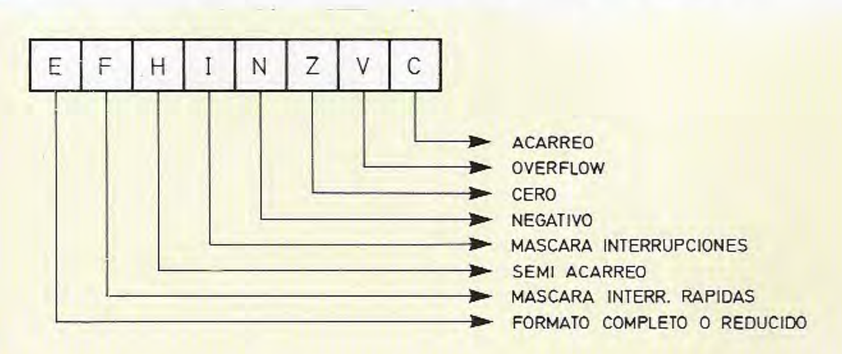
- NMOS: tal es el caso de los DATA GENERAL 601, el DEC LSI 11, el TEXAS TMS 9900, el ZILOG Z-8000 y el MOTOROLA 68000.
- PMOS: el NATIONAL PACE.

Otra tecnología de amplia utilización es la CMOS, la cual se reserva normalmente para la fabricación de versiones de bajo consumo —por ejemplo, para ordenadores portátiles— de los microprocesadores tradicionales.

El número de conexiones o patillas que aparecen en la cápsula que encierra a los principales microprocesadores de 16 bits oscila entre 40 y 64. Cuarenta patillas tienen los microprocesadores DATA GENERAL 601, DEC LSI 11, NATIONAL PACE e INTEL 8086, y 64 los



Organización de los registros internos del microprocesador 6809. Los dos acumuladores de 8 bits pueden utilizarse simultáneamente, disponiendo así de un acumulador único de 16 bits.



El registro de estado del microprocesador 6809 está formado por ocho biestables. Cada uno de ellos almacena un bit de condición, cuyo significado se indica en la figura.

Para saber más

¿Para qué se utilizan los microprocesadores de 16 bits?

Su utilización es muy variada. Podemos destacar, no obstante, dos tipos de usos: como CPU de microprocesadores y como dispositivos de ayuda para equipos más potentes.

¿Qué ventajas fundamentales representan los microprocesado-es de 16 bits?

Un microprocesador genérico de 16 bits presenta frente a los de 8 bits dos mejoras fundamentales: mayor precisión y juego de instrucciones más amplio.

¿Son compatibles los programas y equipos periféricos entre microprocesadores de distinta longitud de palabra?

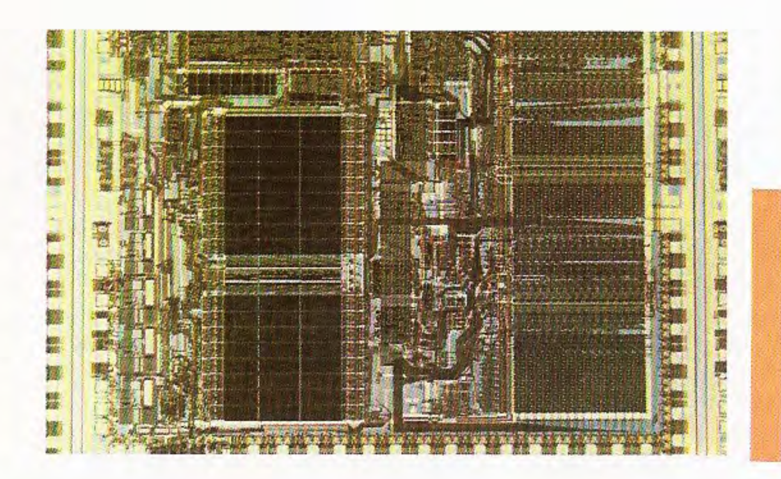
A nivel lógico los microprocesadores de un mismo fabricante suelen ser compatibles. A nivel físico la compatibilidad es, sin embargo, más problemática.

¿Cuántos registros internos tienen los microprocesadores de 16 bits?

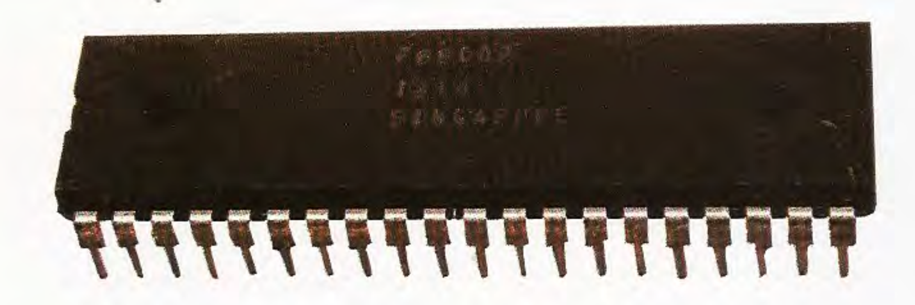
No disponen de un número fijo de registros. La mayoría de los microprocesadores tienen registros de distintos tamaños, que pueden concatenarse formando registros compuestos.

¿Las memorias principales son distintas para los microprocesadores de 8 y de 16 bits?

No. La memoria principal es del mismo tipo. En ambos casos la palabra de memoria es de 8 bits. Los microprocesadores de 16 bits utilizan los datos de memoria concatenados de dos en dos. De esta forma se consigue que el tamaño de la palabra de memoria y de la palabra procesable sea el mismo.



Fotografía del micropro-cesador 6800 de Motorola, ampliada varias decenas de veces. Una vez encapsulado su superficie real ocupará menos de 2 cm².



El microprocesador 6800, una vez encapsulado. El aspecto exterior de este microprocesador de 8 bits es similar al de cualquier otro circuito integrado de 40 patillas.

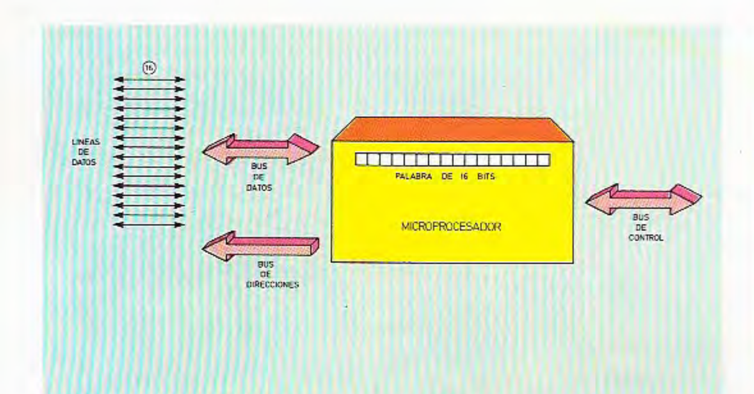
TEXAS TMS 9900 y el MOTOROLA 68000.

Registros internos de los microprocesadores de 16 bits

La mayoría de los microprocesadores de 16 bits disponen de registros internos capaces de resolver distintas tareas. Tal dotación favorece una mayor versatilidad en el uso de los registros, pero también impone la necesidad de especificar su «nombre» en cada instrucción dirigida a la intimidad del microprocesador. Por ejemplo, la instrucción «sumar 1 en el acumulador» no es válida: hay que indicar «sumar 1 en el registro n».

El tamaño de los registros internos de los microprocesadores de 16 bits oscila entre 16 y 32 bits. El 8086 de Intel, por ejemplo, posee catorce registros de 16 bits de uso general, mientras que el 68000 de Motorla incorpora ocho registros de 32 bits para la manipulación de datos y otros ocho registros de 16 bits para el direccionamiento de la memoria.

Otra característica importante de estos microprocesadores es la posibilidad de manipular registros de tamaño distinto al considerado estándar. Los ocho registros de 32 bits del 68000 pueden explotarse según tres formatos distintos: 8, 16 ó 32 bits; mientras que los



Arquitectura interna típica de un microprocesador de 16 bits. La longitud de las palabras determina el tamaño del bus de datos.

	DATA GENERAL 601	DEC L91 11	NATIONAL PACE	TEXAS TMS 9900	INTEL 8086	MOTOROLA 68000	ZILOG Z 8000
TECNOLOGIA	N. MOS	N. MOS	P. MOS	N. MOS	M. MOS	MOS	N. MOS
CAPACIDAD DE MEMORIA	64 K	64 K	64 K	64 K	1 M	16M	8/48 M
CONEXIONES	40	40	40	64	40	64	40/48

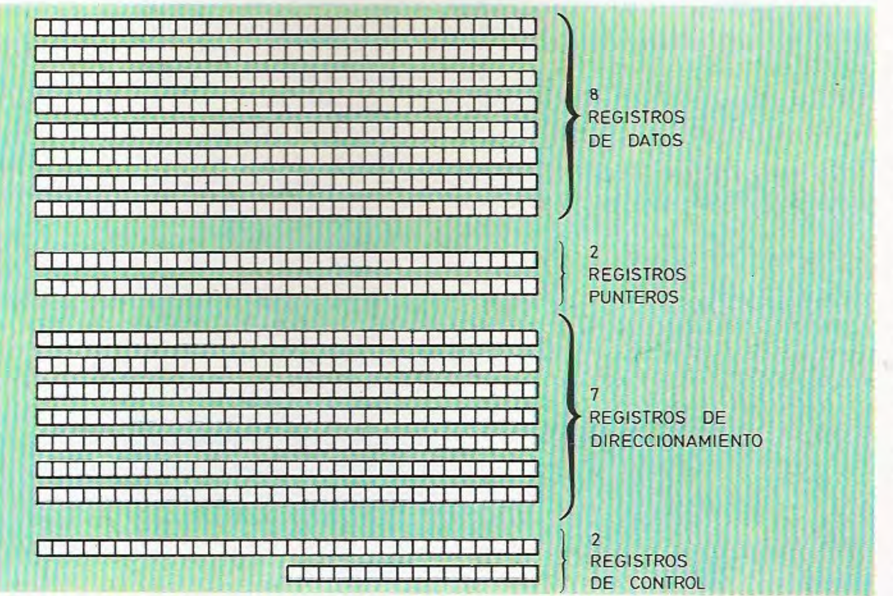
Características generales de los principales microprocesadores de 16 bits. La capacidad de memoria direccionable por cada uno de ellos es función del tamaño del bus de direcciones.

ocho registros de 16 bits sólo pueden utilizarse con el formato de 16 bits.

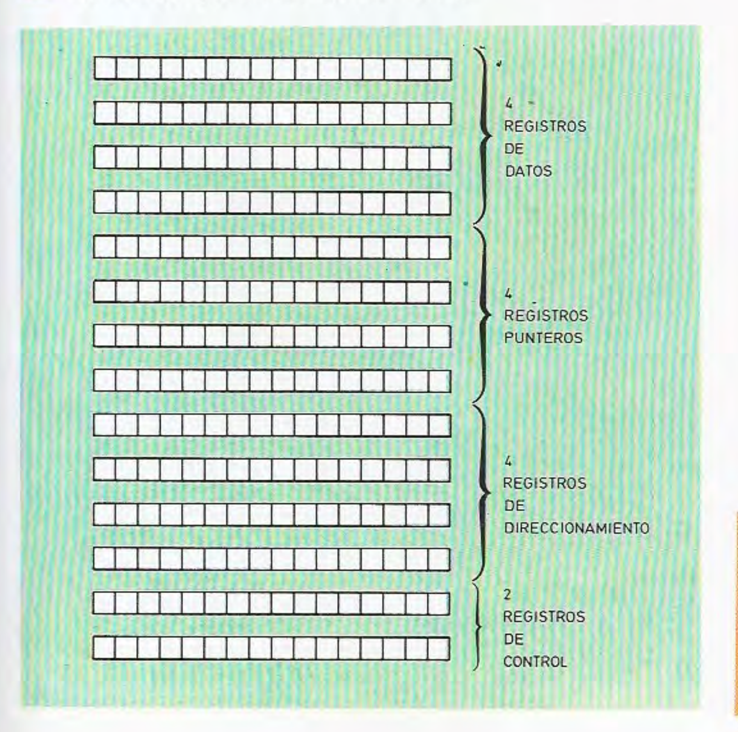
Modos de direccionamiento

Además de las opciones básicas: implícito, inmediato, directo, indirecto, relativo e indexado, cada microprocesador suele disponer de otros modos de direccionamiento especialmente adaptados a su arquitectura interna.

Los sistemas de gestión de memoria implementados por los microprocesadores de 16 bits son esencialmente los mismos que utilizan los chips de 8 bits: para gestionar palabras de 16 bits en una zona de memoria organizada en base a palabras de un byte, se debe apuntar simultáneamente a dos posiciones de memoria.



Todos los registros internos del microprocesador 68000 de Motorola son de 32 bits, excepto el registro de estado, que consta de sólo 16 bits.



Organización interna del microprocesador Intel-8086. Consta de 16 registros de 16 bits cada uno.

Juego de instrucciones

Para medir la potencia del repertorio de instrucciones de un microprocesador no basta con determinar el número de órdenes distintas que es capaz de ejecutar. Tan importante como la cantidad de instrucciones es la complejidad de las operaciones que realizan. Resulta, por tanto, imposible realizar un sumarial estudio comparativo de los juegos de instrucciones, ya que sería necesario detallar todas las operaciones contempladas.

Una cualidad importante de un lenguaje de programación es que las instrucciones sean fáciles de recordar. Esta propiedad no suele ser extensiva a los ensambladores especialmente potentes. Algunos microprocesadores de 16 bits poseen un lenguaje máquina especialmente orientado hacia un determinado lenguaje de programación de alto nivel, generalmente de tipo estructurado, como es el caso del PASCAL.

Interrupciones

El concepto de interrupción de la actividad actualmente en curso por parte del microprocesador, resulta palpable en aquellas ocasiones en las que es necesario detener la ejecución para realizar alguna tarea más urgente. El microprocesador debe ser capaz, cuando dicha tarea haya finalizado, de continuar la ejecución del programa interrumpido.

La mayoría de los microprocesadores de 16 bits reservan una zona de memoria en la que almacenan un vector con todas las posibles situaciones de interrupción. Cuando el usuario ordena proseguir la ejecución de un programa previamente interrumpido, el microprocesador se limita a comprobar el estado del vector de interrupciones para determinar las operaciones a realizar para continuar con la ejecución.

Memoria virtual

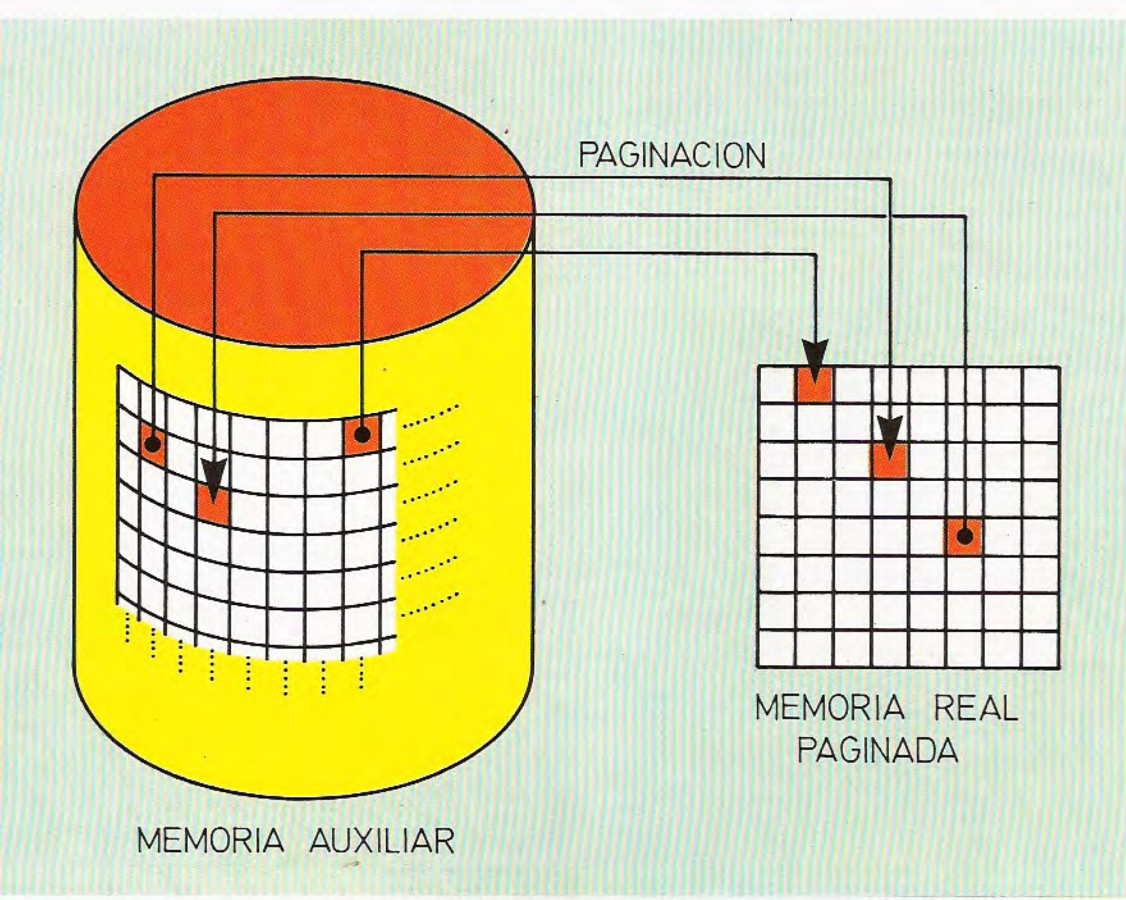
La memoria central de un ordenador puede dividirse en partes de igual tamaño a las que se denomina páginas. Se llama paginación a la técnica que permite acceder a un mayor número de páginas de las que caben en la memoria central. Se consigue con ello que la capacidad de direccionamiento sea superior a las posiciones de memoria direccionables.
Los tres objetivos principales de las técnicas de paginación son los siguientes:

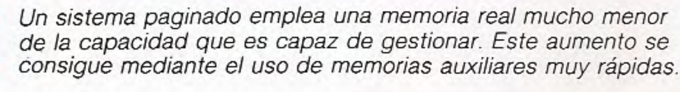
- Evitar el fraccionamiento de la memoria central.
- Simular una máquina virtual de memoria ilimitada.
- Permitir la utilización del ordenador a varios usuarios en la modalidad que se denomina de tiempo compartido.

El principio de funcionamiento de las máquinas paginadas se basa en que todo usuario del ordenador dispone para la ejecución de sus programas de una memoria virtual, cuyo tamaño está determinado por la capacidad de direccionamiento de las instrucciones y no por la capacidad de memoria real. Se utiliza para ello memoria auxiliar en la que se almacenan las páginas de memoria no utilizadas en un instante dado.

El programador no se preocupa de gestionar la memoria auxiliar; trata la memoria virtual de que dispone como si fuese memoria central, a la que se denomina también memoria real.

Aunque el operador tan sólo puede acceder a la información contenida en la memoria real, cuando un programa solicita la intervención de una dirección de memoria virtual, el sistema se encarga de realizar la copia de la página que contiene la dirección solicitada desde la memoria auxiliar hasta la memoria real.
El sistema operativo invierte un determinado tiempo en realizar las transferencias de páginas entre las memorias auxiliar y principal; si el cociente memoria virtual/memoria real es demasiado grande, el tiempo que tarda en gestionar la memoria virtual es tan alto que la ejecución de los programas del usuario queda prácticamente paralizada. La capacidad del ordenador y la velocidad de gestión de éste determina el número máximo de páginas que se pueden transferir en un segundo, y este número, a su vez, la máxima memoria virtual que puede ser gestionada eficazmente por el sistema operativo.




Un sistema paginado emplea una memoria real mucho menor de la capacidad que es capaz de gestionar. Este aumento se consigue mediante el uso de memorias auxiliares muy rápidas.

Circuitos lógicos

Sistemas combinacionales y secuenciales



Los cimientos de la informática se encuentran en la ciencia electrónica; más precisamente en el área de los circuitos lógicos o circuitos digitales.

A lo largo del presente capítulo se introducirán los conceptos básicos de la circuitería lógica o digital.

Lógica de niveles

Los circuitos lógicos trabajan con dos niveles de tensión diferenciados: alto y bajo. Según la relación que se establezca entre estos niveles y los valores lógicos «1» y «0», tendremos dos lógicas distintas:

- *Lógica positiva:* representa el estado lógico 1 mediante la tensión más elevada y el estado 0 mediante la tensión más baja.
- *Lógica negativa:* representa el estado lógico 1 mediante la tensión más baja y el estado 0 mediante la tensión más elevada.

Normalmente el nivel de tensión más bajo es el de referencia, esto es: 0 voltios. Ello no es obligatorio, ya que pueden asignarse dos voltajes cualquiera a los dos estados binarios; por supuesto, siempre que ambos niveles queden perfectamente diferenciados.

Tablas de verdad

Una tabla de verdad está constituida por dos zonas: la de entrada y la de salida.

- *Zona de entrada*

Contiene todas las posibles combinaciones lógicas que se pueden presentar a la entrada del circuito.

- *Zona de salida*

Al aplicar la tabla de verdad a los circuitos lógicos, esta zona contiene los valores de salida correspondientes a cada posible entrada al circuito.

La zona de entrada se suele representar a la izquierda de la tabla y la de salida, a la derecha. En la cabecera aparece la identificación de la variable representada en cada columna, tanto en la zona de entrada como en la de salida.

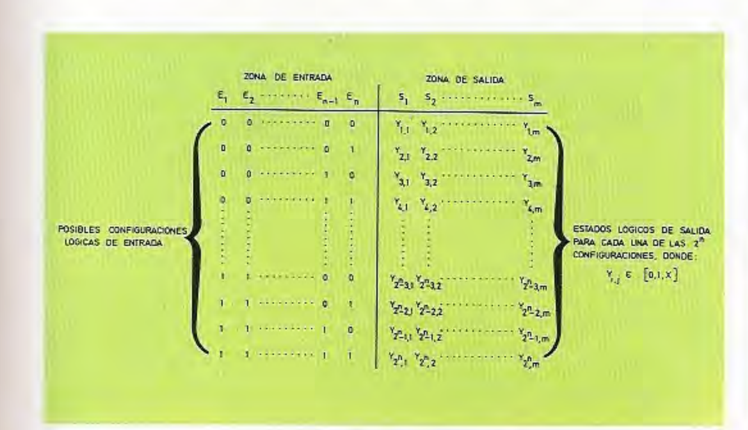
Cada fila de la tabla debe contener tantos valores como columnas se utilicen. Dichos valores sólo podrán ser: 0, 1 o X. Mediante los dos primeros se representan los estados lógicos bajo y alto; el tercero (X) corresponde a estado lógico indiferente (no precisado). Así, por ejemplo, si en una posición de la tabla aparece una «X», el valor que tome la variable en cuestión no influye en la combinación lógica de la línea en la que aparezca.

Cronogramas

Un cronograma consiste en una representación gráfica de los estados de una o varias funciones lógicas respecto al tiempo. Para ello se utiliza el eje de abscisas (horizontal) para la variable tiempo y el eje de ordenadas (vertical) para los estados lógicos (tensiones) observados en cada intervalo de tiempo.

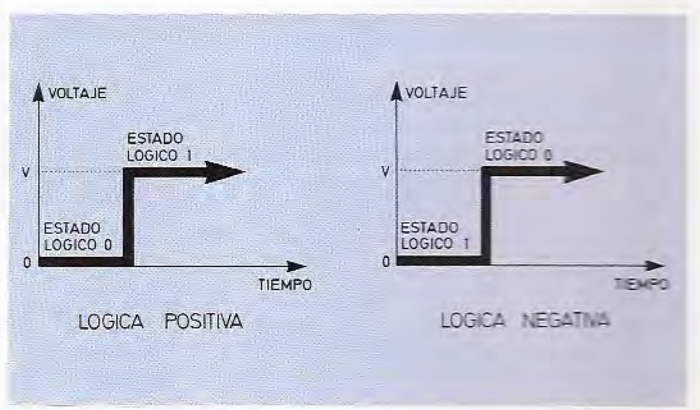
Si el cronograma contiene información de más de una función, se pueden representar unas bajo las otras. En la parte superior del gráfico se incluye la señal patrón de entrada. Normalmente esta señal patrón es generada por el «reloj» que sincroniza al conjunto de todas las funciones.

Para ilustrar los conceptos estudiados, vamos a representar a continuación la tabla de verdad y el cronograma correspondiente a un circuito cuya única entrada es una señal de reloj y que entrega una salida con dos estados complementarios, alterándolos en cada ciclo de reloj.

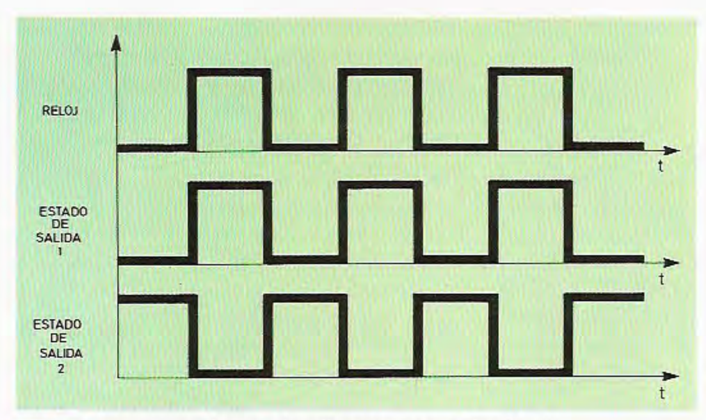


ZONA DE ENTRADA					ZONA DE SALIDA			
E_1	E_2	········	E_{n-1}	E_n	S_1	S_2	········	S_m
0	0	········	0	0	$Y_{1,1}$	$Y_{1,2}$	········	$Y_{1,m}$
0	0	········	0	1	$Y_{2,1}$	$Y_{2,2}$	········	$Y_{2,m}$
0	0	········	1	0	$Y_{3,1}$	$Y_{3,2}$	········	$Y_{3,m}$
0	0	········	1	1	$Y_{4,1}$	$Y_{4,2}$	········	$Y_{4,m}$
⋮	⋮		⋮	⋮	⋮	⋮		⋮
1	1	········	0	0	$Y_{2^n-3,1}$	$Y_{2^n-3,2}$	········	$Y_{2^n-3,m}$
1	1	········	0	1	$Y_{2^n-2,1}$	$Y_{2^n-2,2}$	········	$Y_{2^n-2,m}$
1	1	········	1	0	$Y_{2^n-1,1}$	$Y_{2^n-1,2}$	········	$Y_{2^n-1,m}$
1	1	········	1	1	$Y_{2^n,1}$	$Y_{2^n,2}$	········	$Y_{2^n,m}$

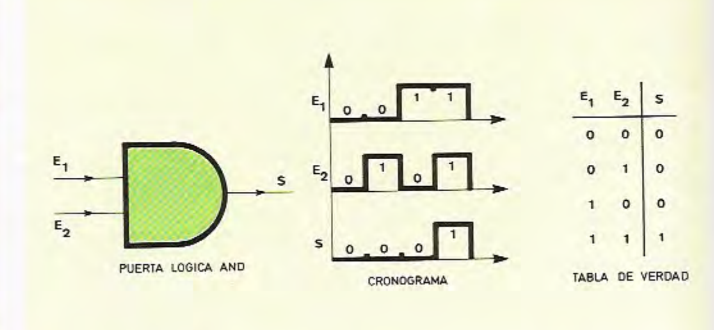
Tabla de verdad de un circuito lógico. A la izquierda se representan todas las posibles combinaciones de unos y ceros que pueden aparecer en la entrada del circuito; la zona derecha refleja los estados lógicos de salida que corresponden a cada combinación de entrada.



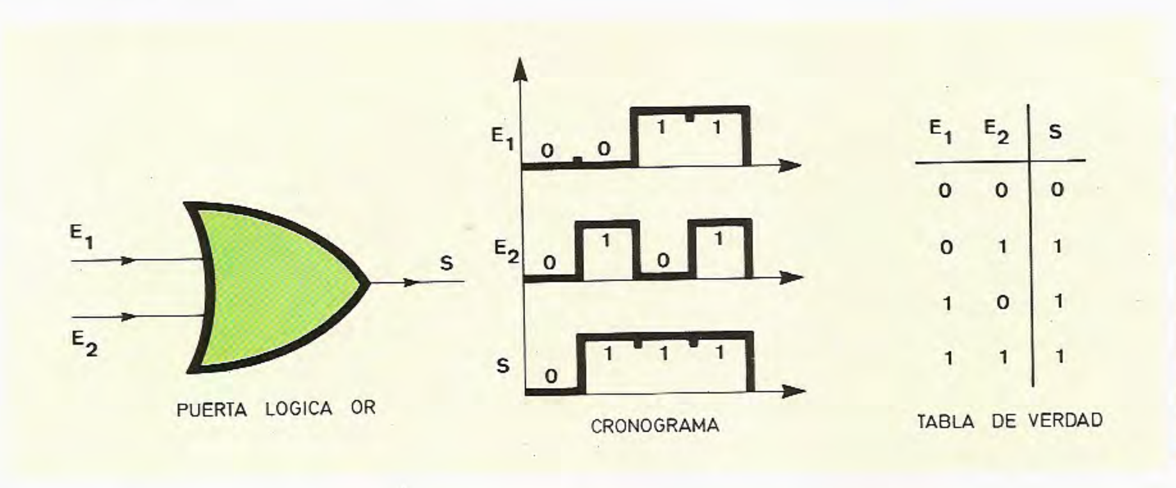
En lógica positiva se asigna el estado lógico «1» al nivel de tensión más alto. Por el contrario, en lógica negativa, el «1» lógico se corresponde con el nivel de tensión inferior.



Cronograma de un circuito con una entrada de reloj y dos salidas. Los valores lógicos de las salidas son complementarios.



La puerta lógica AND efectúa la operación «producto lógico». La salida estará a nivel lógico alto sólo cuando se aplique un estado alto a ambas entradas.



Puerta lógica OR. Realiza la función «suma lógica». La salida del circuito adquiere estado lógico alto, cuando se aplica un estado alto en cualquiera de las dos entradas.

Circuitos combinacionales

Denominamos circuito combinacional a la representación de una función booleana mediante los operadores: suma lógica ($\vee$), producto lógico ($\wedge$), complementación (—) y suma lógica exclusiva ($\oplus$). Por ejemplo, la función booleana $f(X_1, X_2) = (X_1 \vee X_2, X_1 \wedge X_2)$ puede representarse mediante una tabla de verdad, un cronograma y un bloque funcional del circuito, según se ilustra en el gráfico correspondiente.

Puertas lógicas

Cuando un circuito realiza una función booleana con varias variables de entrada y entregando una única salida, se dice que es una puerta lógica. El circuito que ilustra el ejemplo anterior tiene dos salidas: no es, por tanto, una puerta lógica. A continuación vamos a describir las puertas básicas u operadores lógicos elementales, los cuales nos servirán para diseñar circuitos más complejos.

- *Suma lógica (OR)*

La función lógica OR responde a la tabla de verdad y cronograma que se representa en la figura correspondiente.

Si una cualquiera de las dos variables de entrada toma el valor uno, el resultado de la operación será uno. En el caso de que las dos entradas valgan cero, el resultado será cero.

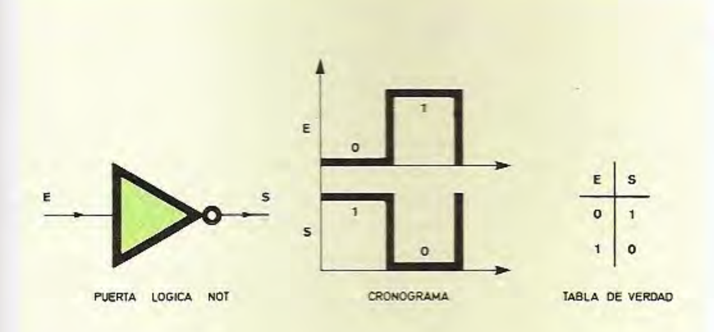
En lógica negativa, el funcionamiento de este operador coincide con el del operador producto lógico. Para comprobarlo basta con cambiar los ceros por unos, y viceversa, en la tabla de verdad.

- *Producto lógico (AND)*

En la figura correspondiente se puede observar la tabla de verdad que define a esta operación lógica, su cronograma y el símbolo que la representa.

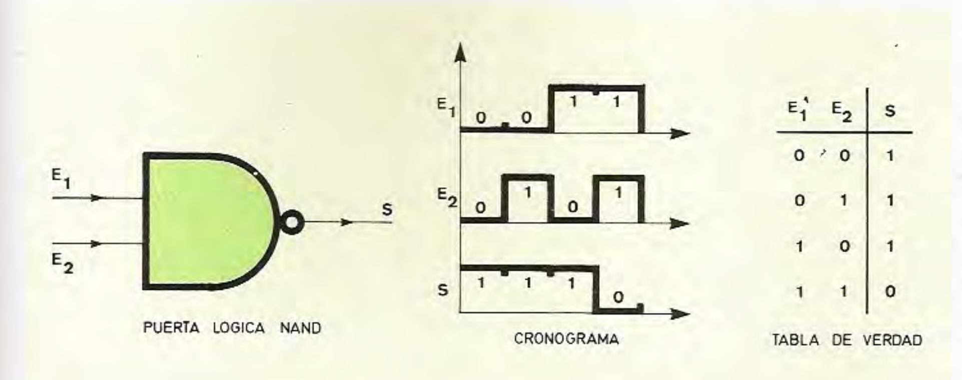
El resultado de su aplicación a dos variables es uno si, y sólo si, ambas tomas el valor uno.

En lógica negativa la función desarrollada por la operación AND es idéntica a la resultante de la operación OR en lógica positiva.



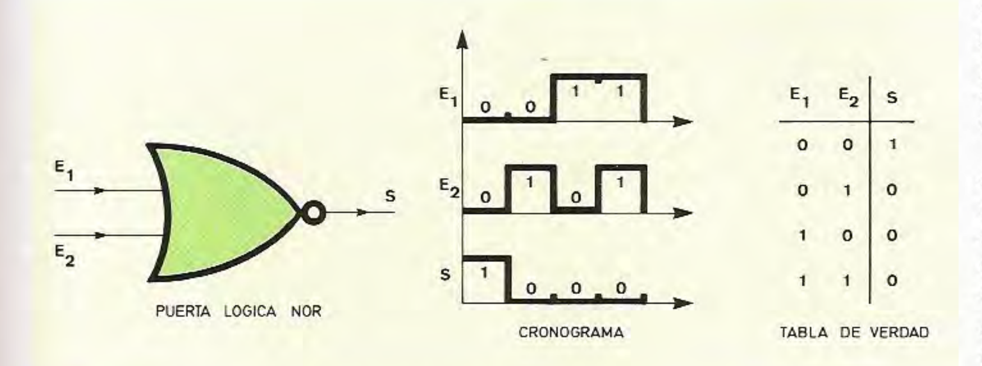
E	S
0	1
1	0

La función lógica de complementación, suele también denominarse negación o inversión. La salida de la puerta NOT entrega el estado lógico opuesto al aplicado en la entrada.



E_1	E_2	S
0	0	1
0	1	1
1	0	1
1	1	0

La puerta lógica NAND realiza la operación de producto lógico complementario de los estados de entrada.



E_1	E_2	S
0	0	1
0	1	0
1	0	0
1	1	0

La puerta lógica NOR sintetiza la suma lógica complementada de los estados lógicos aplicados a las líneas de entrada.

- *Complementación (NOT)*

La tabla de verdad de esta función lógica tan sólo consta de una columna en la zona de entrada y otra en la de salida. En la ilustración se ha representado su tabla de verdad, su cronograma y el símbolo asociado. Si la entrada es uno, el resultado será cero; por el contrario, si la entrada es cero, el resultado será uno.

En algunos textos esta operación lógica se denomina inversión o negación, pero en cualquier caso se representa mediante una barra que afecta a la variable o expresión a complementar.

- *Suma negada (NOR)*

Esta puerta lógica produce el resultado contrario al de la puerta lógica OR (suma lógica). En su tabla de verdad se pueden comprobar los resultados de esta operación. Aplicar la operación NOR a dos variables equivale a sumarlas lógicamente y calcular luego el complementario del resultado.

- *Producto negado (NAND)*

El resultado que obtiene este operador es idéntico al producido por la actuación consecutiva de los operadores producto lógico y complementación. En la figura correspondiente se representa la tabla de verdad que refleja su funcionamiento, el símbolo con que se representa y el cronograma correspondiente.

- *Suma lógica exclusiva (OR-EXCLUSIVA)*

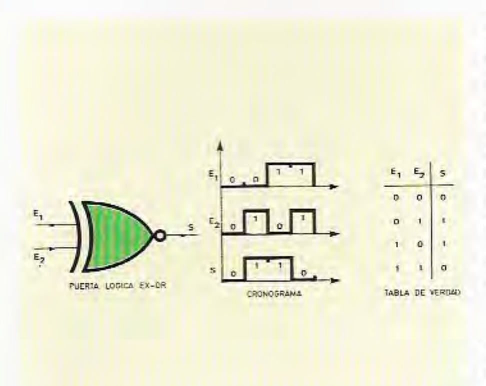
La función lógica OR-EXCLUSIVA opera según la tabla de verdad que muestra la ilustración.

Su funcionamiento es parecido al de la operación suma lógica (OR), pero exige que una, y sólo una de las dos variables sobre las que opera, tome el valor uno para que el resultado sea uno; en cualquier otro caso el resultado producido es cero.

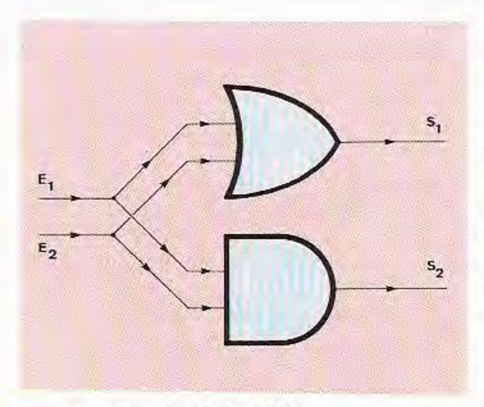
Las operaciones lógicas elementales son: suma lógica, producto lógico y complementación. Cualquier otra operación de las que hemos definido se puede sustituir por una combinación de las operaciones elementales. Por ejemplo: $a \text{ NOR } b \equiv \overline{a \text{ OR } b}$; $a \text{ NAND } b \equiv \overline{a \text{ AND } b}$, y a OR-EXCLUSIVA $b \equiv (a \text{ AND } \bar{b}) \text{ OR } (\bar{a} \text{ AND } b)$. La comprobación de estas identidades se puede realizar aplicando el procedimiento usual para demostrar igualdades lógicas.

Circuitos secuenciales

A continuación iniciaremos el estudio de los denominados circuitos lógicos



La puerta lógoca OR-Exclusiva efectúa la operación de suma lógica exclusiva; el estado lógico de salida será alto cuando las dos entradas reciban estados lógicos distintos.



La función lógica f(X1, X2) = (X1vX2, X1∧X2) puede operarse por medio de un circuito combinacional formado por dos puertas lógicas: la OR calcula S1 = X1vX2, y la AND el valor S2 = X1∧X2.

secuenciales, partiendo de la descripción de los principales elementos biestables. Los circuitos secuenciales son circuitos lógicos combinacionales, aunque «con memoria»; es decir, son capaces de almacenar información: los valores de salida del circuito no dependen sólo de las entradas, sino también de ciertos datos almacenados en su interior.

Dentro de todo circuito lógico secuencial se almacenan informaciones en forma de niveles lógicos. Cada dato almacenado se denomina también estado del circuito. De esta forma, los valores de salida en un instante dado son función de los valores de entrada y del estado del circuito en ese instante.

Cada vez que se produce una nueva entrada en el circuito, éste adopta un estado lógico que depende de la nueva entrada y de su estado anterior.

Elementos biestables

Un biestable es un circuito lógico secuencial capaz de almacenar un elemento de información binaria; dado que los dígitos binarios son dos (0 y 1), el elemento biestable tendrá dos posibles estados lógicos internos (alto: «1», o bajo: «0»).

Un biestable suele disponer de una, dos o, en ocasiones, hasta cuatro entradas. En función del valor de estas entradas y del valor lógico almacenado en su interior, adopta un nuevo estado interno y entrega dos salidas: Q, que coincide con el estado interno anterior, y $\overline{Q}$, que coincide con el valor lógico complementario de Q.

Los elementos biestables pueden funcionar en modo asíncrono o síncrono. En modo síncrono un reloj exterior se encarga de sincronizar o controlar su funcionamiento. Las entradas que llegan a un biestable síncrono sólo serán admitidas durante el intervalo en el que esté activada la señal del reloj.

A continuación vamos a describir los principales tipos de biestables:

- *Biestable T* (Trigger-disparo)

En modo asíncrono este biestable tiene una única entrada. Cuando dicha entrada toma el valor 1 (disparo), el estado lógico interno cambia automáticamente de valor. Convencionalmente, adoptaremos la siguiente nomenclatura:

T = Estado lógico de entrada.
Q_1 = Estado interno antes de la entrada T.
Q_{t+1} = Nuevo estado lógico de salida.

El funcionamiento del biestable T, en modo asíncrono, viene reflejado por la tabla de verdad de la figura correspondiente. El estado interno permanece fijo ($Q_{t+1} = Q_1$ puede representarse) cuando la entrada T es cero y cambia de valor ($Q_{t+1} = Q_t$) cuando la entrada es uno. Por tanto, la función lógica que realiza el biestable T será la siguiente:

$$Q_{t+1} = (\overline{T} \wedge Q_t) \vee (T \wedge \overline{Q}_t) = T \oplus Q_t$$

Cuando a la entrada de un biestable T asíncrono adaptamos la salida de una puerta lógica AND, cuyas entradas son la variable de entrada del circuito y una señal de reloj, obtenemos un biestable T síncrono.

En este caso, el estado interno del biestable sólo cambia de valor cuando tanto la entrada como la señal de reloj tomen el valor lógico «1».

- *Biestable R-S* (Reset-Set)

Las dos entradas de este biestable se denominan R y S. Cuando en la entrada R aparece un 1 lógico, el estado interno cambia a 0 (Reset). Si el 1 lógico llega a la entrada S, el estado interno cambia a nivel alto (Set). Cuando las dos entradas R y S toman el valor 0, el estado interno no se altera. La aplicación

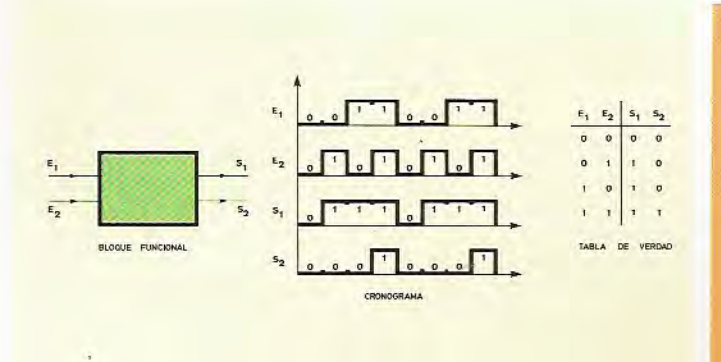


Tabla de verdad y cronograma de la función lógica f(X1, X2) = (X1vX2, X1∧X2). El bloque funcional que la representa tiene dos salidas y, por tanto, no constituye una puerta lógica.

de un estado lógico alto (1) a ambas entradas R y S es una posibilidad no admitida en los biestables de este tipo.

A partir de la tabla de verdad reflejada en el gráfico correspondiente a este biestable, deducimos la función lógica del mismo:

$$Q_{t+1} = S \vee (Q_t \wedge \overline{R})$$

Para conseguir el funcionamiento sincronizado de este biestable son necesarias dos puertas lógicas AND. A una de ellas ingresan la señal de reloj y la línea R; en la otra puerta se introducen la misma señal de reloj y la línea S. Las salidas de ambas puertas se conectan a las entradas del biestable R-S. De esta forma, para que llegue un uno lógico a cualquiera de las líneas de entrada del biestable, la señal de reloj debe estar a nivel lógico alto.

- *Biestable J-K*

Este nuevo elemento lógico biestable reúne las características de los dos anteriores: T y R-S. El funcionamiento es similar al del biestable R-S, sin más que identificar el valor de S con el de J y el de R con el de K. En este caso se permite la aplicación simultánea de dos unos lógicos a las entradas J y K. Esta configuración (J = 1, K = 1) origina la inversión automática del estado lógico interno.

La tabla de verdad característica de este elemento biestable y su símbolo lógico están representados en el gráfico correspondiente. De la tabla de verdad se deduce fácilmente la función lógica del biestable J-K:

$$Q_{t+1} = (\overline{Q}_t \wedge J) \vee (Q_t \wedge \overline{K})$$

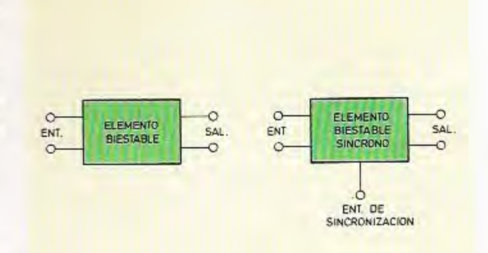
Para que el elemento J-K funcione en modo síncrono hay que añadirle una entrada de reloj, a través de dos puertas AND, tal y como se realizó con el biestable de tipo R-S.

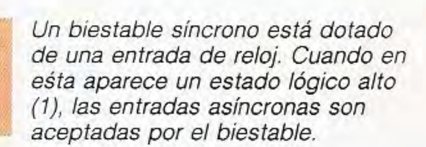
- *Biestable D* (Delay)

La misión encomendada a este biestable es retardar un determinado período de tiempo la información que reciba. Si opera en modo asíncrono, el estado lógico de la línea de entrada aparecerá en la línea de salida con un tiempo de demora igual al tiempo de conmutación del biestable. Si, por el contrario, se trabaja en modo síncrono, la información no pasará al interior del biestable hasta que la señal del reloj tome el valor uno. En ambos casos, el dato de entrada se almacena hasta que aparezca un nuevo estado lógico en la línea D.

De la tabla de verdad se deduce que la función lógica que lo caracteriza es la siguiente:

$$Q_{t+1} = D$$




Un biestable síncrono está dotado de una entrada de reloj. Cuando en ésta aparece un estado lógico alto (1), las entradas asíncronas son aceptadas por el biestable.

$Q_{T+1} = T \oplus Q_T$

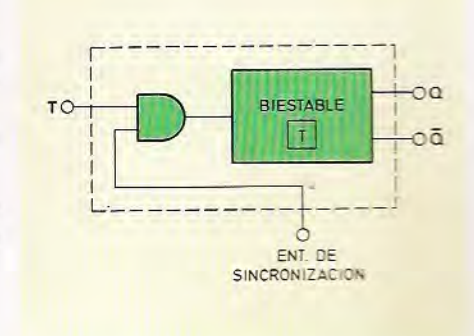
T	Q_T	Q_{T+1}
0	0	0
0	1	1
1	0	1
1	1	0

Representación gráfica, tabla de verdad y función lógica de un biestable de tipo T. Cuando la entrada T toma el valo lógico 1 el estado interno cambia.

Comparación entre biestables

Con objeto de abaratar la producción de circuitos lógicos es conveniente utilizar el mínimo número de tipos de biestables. Para ello se ha elegido un biestable base que, con la simple inclusión de alguna puerta lógica o de alguna conexión especial, permite sintetizar cualquier otro tipo de biestable.

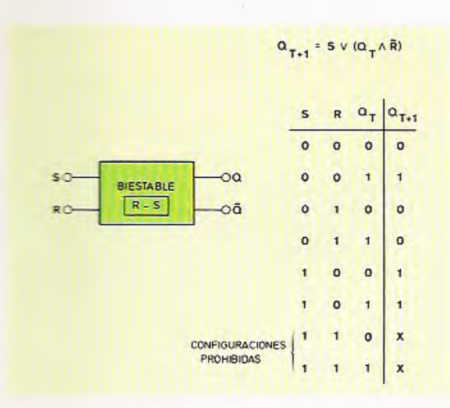
Comparando las características de los biestables estudiados anteriormente, observamos que el tipo J-K puede servir de base para sintetizar cualquiera de los restantes. A continuación vamos a estudiar la forma en la que se obtiene cualquier otro tipo de biestable a partir de un elemento del tipo J-K.



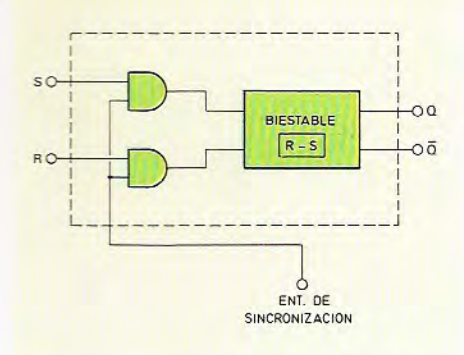
Un biestable T asíncrono, a cuya entrada accede la salida de una puerta lógica AND, funciona en modo síncrono. La señal de reloj y la señal T se aplican a las entradas de la citada puerta lógica.

- *Biestable T*

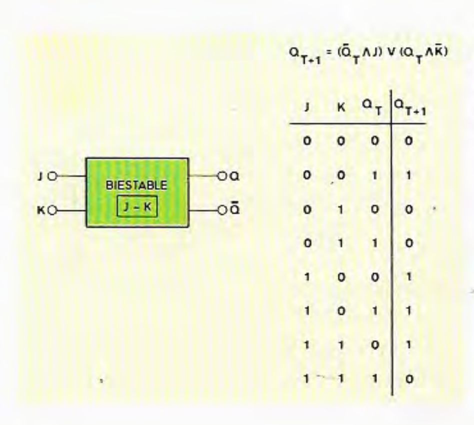
Cuando las dos entradas de un biestable J-K reciben un estado lógico bajo (0), su estado interno se conserva. Si ambas reciben un 1 lógico, el estado interno cambia de valor. Si unimos las dos entradas J-K de forma que ambas tomen siempre el mismo valor, uno o cero, nos encontraremos ante un biestable T. La entrada T coincidirá con el punto común de las líneas J-K.



Símbolo de representación de un biestable R-S asíncrono, tabla de verdad y función lógica que realiza. La configuración de entrada S = 1 y R = 1 no está autorizada, dado que llevaría al flip-flop a un funcionamiento imprevisible.



Realización práctica de un biestable R-S síncrono. Las entradas del biestable asíncrono son atacadas a través de sendas puertas lógicas AND.



Biestable J-K asíncrono. Su tabla de verdad coincide con la propia del biestables R-S, salvo en la configuración de entrada J = 1 y K = 1, que en este caso está permitida.

Tipos de biestables

Biestable	Descripción	Entradas	Función característica
TRIGGER (disparo) «T»	El objeto de este biestable consiste únicamente en actualizar el estado interno en función de una única entrada. Si ésta es «0», el estado anterior se conserva. Si es «1», se cambia por el valor complementario del estado interno anterior.	1 (T)	$Q_{t+1} = (\overline{T} \wedge Q_t) \vee (T \wedge \overline{Q}_t)$
RESET-SET «R-S»	Si el valor de entrada R es «1», el nuevo valor del estado interno será «0». Cuando la entrada S es «1», el nuevo valor será «1». En el caso de que las dos entradas valgan «0», el estado interno no se modifica.	2 (R, S)	$Q_{t+1} = S \vee (Q_t \wedge \overline{R})$
«J-K»	La función de este biestable es una mezcla de la de los dos anteriores. Para funcionar como el biestable «R-S» se hace corresponder la entrada S con la J y la entrada R con la K. Para simular al biestable T utiliza las configuraciones con ambas entradas a «1».	2 (J, K)	$Q_{t+1} = (\overline{Q}_t \wedge J) \vee (Q_t \wedge \overline{K})$
DELAY (retardo) «D»	El biestable «D» sirve únicamente para retardar la salida que le llegue; por tanto, el estado interno toma siempre el valor de la última entrada.	1 (D)	$Q_{t+1} = D$

• *Biestable R-S*

La tabla de verdad de un biestable R-S es idéntica a la del biestable J-K, a excepción de la combinación de entrada compuesta por dos unos, prohibida en el R-S.

En definitiva, para simular el comportamiento del biestable R-S a partir de uno J-K, basta con tomar la entrada J como S y la entrada K como R.

• *Biestable D*

El funcionamiento del biestable D es el más sencillo de todos: su actuación se reduce a retardar la información binaria de entrada. Cuando su única línea de entrada, D, recibe un estado lógico bajo (cero), el siguiente estado interno que toma el circuito y la salida serán también cero lógico. Cuando D vale uno, el siguiente estado interno y la salida pasarán a uno lógico.

Si observamos las tablas de verdad de los biestables D y J-K podemos comprobar que en todas las configuraciones de entrada se cumple que D = J y $\bar{D}$ = K. Por tanto, para simular el funcionamiento del biestable D mediante un biestable J-K es suficiente introducir los datos a través de J, y con incluir una puerta lógica NOT entre la entrada de datos y K, de forma que entre por ésta el valor $\bar{D}$.

Aunque nuestro esfuerzo se ha centrado en simular todos los tipos de biestables a partir de un dispositivo J-K, podríamos haber tomado como base los flip-flops R-S o T. No es nuestro objetivo realizar una nueva construcción semejante a la ya descrita, por lo que dejamos al lector la tarea de obtener las correspondientes conversiones.

Diseño de circuitos lógicos

Las técnicas actuales para la fabricación de circuitos integrados a gran esca-



El código ASCII

Los ordenadores no sólo se ocupan de manipular cifras numéricas; también deben tratar información compuesta por letras del alfabeto y por algunos signos especiales, debidamente codificados. Para representar estos caracteres en el interior de la máquina se ha adoptado, a nivel internacional, el código alfanumérico ASCII («American Standard Code for Information Interchange» o, lo que es lo mismo, código estándar americano para el intercambio de información).
Inicialmente se desarrolló un código ASCII basado en 6 bits que permitía representar $2^6 = 64$ configuraciones distintas (10 caracteres numéricos decimales, 26 caracteres alfabéticos y 28 símbolos especiales). Posteriormente se aumentó a 7 el número de bits utilizados, con lo que se llegaron hasta $2^7 = 128$ configuraciones distintas, que incluían los mismos caracteres alfanuméricos de la primera versión, más algunas expresiones de control y símbolos ortográficos. Hoy se opera incluso con códigos ASCII de 8 bits, que permiten representar hasta 256 caracteres, símbolos y códigos de control.
La tabla adjunta representa el código ASCII de 7 bits. Para su interpretación hay que tener en cuenta que los bits que conforman la palabra binaria representativa de cada carácter codificado se ordenan de derecha a izquierda, tal como se indica a continuación:

6	5	4	3	2	1	0

CODIGO ASCII DE 7 BITS

Bits 3210	Bits 654							
	000	001	010	011	100	101	110	111
0000	NUL	DLE	SP	0	@	P	'	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	'	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	g	I	Y	i	v
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	/	l	¦
1101	CR	GS	—	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	—	o	DEL

SIGNIFICADO DE LAS ABREVIATURAS

Abreviatura	Significado
NUL	Nulo.
SOH	Principio de encabezamiento.
STX	Comienzo de texto.
ETX	Fin de texto.
EOT	Fin de transmisión.
ENQ	Pregunta.
ACK	Acuse de recibo.
BEL	Timbre (señal).
BS	Retroceso.
HT	Tabulación horizontal.
LF	Cambio de renglón.
VT	Tabulación vertical.
FF	Página siguiente.
CR	Retroceso de carro.
SO	Fuera de código.
SI	En código.
SP	Espacio.
DLE	Encaje de transmisión.
ESC	Escape.
DEL	Supresión.
DC1	Mando de dispositivo auxiliar 1.
DC2	Mando de dispositivo auxiliar 2.
DC3	Mando de dispositivo auxiliar 3.
DC4	Mando de dispositivo auxiliar 4.
NAK	Acuse de recibo negativo.
SYN	Sincronización de reposo.
ETB	Fin de bloque de transmisión.
CAN	Cancelación.
EM	Fin de medio físico.
SUB	Sustitución.
FS	Separador de fichero.
GS	Separador de grupo.
RS	Separador de registro.
US	Separador de unidad.

Significado de las abreviaturas en el código ASCII completo de 7 bits.

Código EBCDIC

A pesar de que el código ASCII es el más difundido para la representación de caracteres alfanuméricos, algunas compañías de sistemas informáticos han creado y utilizan en ciertos casos su propio código.
Después del ASCII, el código alfanumérico más habitual es el desarrollado por la firma IBM: el denominado EBCDIC. Este código utiliza 8 bits y deja algunas de las $2^8 = 256$ configuraciones posibles sin asignación generalizada. La ordenación de los bits se ha tomado, al igual que en el código ASCII, de derecha a izquierda, tal como se indica a continuación:

7	6	5	4	3	2	1	0

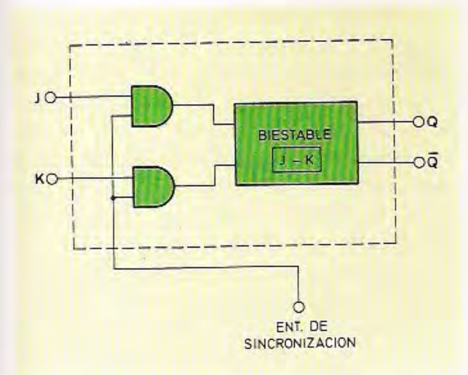
CODIGO EBCDIC (IBM)

Bits 3210	Bits 7654															
	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0000	NUL	DLE	DS		SP	—							{	}	\	
0001	SOH	DC1	SOS						a	j	~		A	J		1
0010	STX	DC2	FS	SYN					b	k	s		B	K	S	2
0011	ETX	DC3							c	l	t		C	L	T	3
0100	PF	RES	BYP	PN					d	m	u		D	M	U	4
0101	HT	NL	LF	RS					e	n	v		E	N	V	5
0110	LC	BS	EOT/ETB	VC					f	o	w		F	O	W	6
0111	DEL	IL	PRE/ESC	EOT					g	p	x		G	P	X	7
1000		CAN							h	q	y		H	Q	Y	8
1001	RLF	EM						`	i	r	z		I	R	Z	9
1010	SMM	CC	SM		,	!	*	:								
1011	VT			<	.	$	,	#								
1100	FF	IFS		DC4		.	%	@								
1101	CR	IGS	ENQ	NAK	(	)	—	|								
1110	SO	IRS	ACK		+	;	>	=								
1111	SI	IOS	BEL	SUB		¬	?	″								

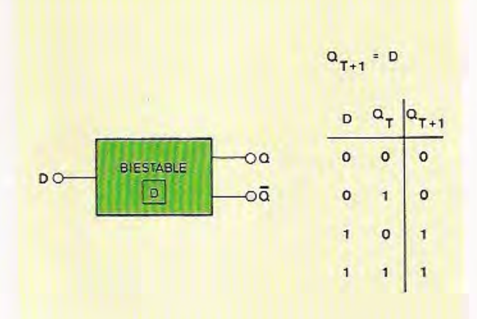
SIGNIFICADO DE LAS ABREVIATURAS

Abreviatura	Significado	Abreviatura	Significado	Abreviatura	Significado
NUL	Nulo.	DC3	Mando de dispositivo auxiliar 3.	FS	Separador de fichero.
SOH	Principio de encabezamiento.	RES	Reposición.	BYP	Desviación.
STX	Comienzo de texto.	NL	Nueva línea.	LF	Cambio de renglón.
ETX	Fin de texto.	BS	Retroceso.	EOB/ETB	Fin de bloque/Fin del bloque de transmisión.
PF	Final de perforación.	IL	Inactivo.	PRE/ESC	Prefijo escape.
HT	Tabulación horizontal.	DC4	Mando de dispositivo auxiliar 4.	SM	Poner modo.
LC	Minúsculas.	NAK	Acuse de recibo negativo.	ENQ	Pregunta.
DEL	Supresión.	CAN	Anulación.	ACK	Acuse de recibo.
RLF	Cambio inverso de renglón.	EM	Fin del medio físico.	BEL	Timbre.
SMM	Comienzo de mensaje manual.	CC	Control del cursor.	SYN	Sincronizado de reposo.
VT	Tabulación vertical.	IFS	Cambio de separador de fichero.	PN	Comienzo de perforación.
FF	Página siguiente.	IGS	Cambio de separador de grupo.	RS	Separador de registro.
CR	Retroceso del carro.	IRS	Cambio de separador de registro.	VC	Mayúsculas.
SO	Fuera de código.	IVS	Cambio de separador de unidad.	EOT	Fin de transmisión.
SI	En código.	DS	Selección del dígito.	SUB	Sustitución.
DLE	Encaje de transmisión.	SOS	Comienzo de significado.	SP	Espacio.
DC1	Mando de dispositivo auxiliar 1.				
DC2	Mando de dispositivo auxiliar 2.				

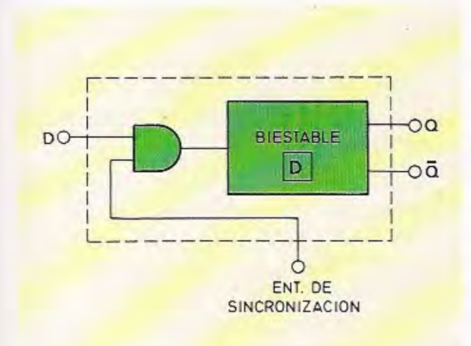
Significado de las abreviaturas utilizadas en el código EBCDIC.



La transformación de un biestable J-K asíncrono en otro J-K síncrono se realiza de forma idéntica a la aplicada en un biestable R-S: adaptando dos puertas AND a las entradas asíncronas.



El biestable más simple es el de tipo D: la señal de entrada se transmite a la salida con un determinado tiempo de retardo.



Para sincronizar un biestable D hay que acceder a la entrada D a través de una puerta lógica AND que opera la señal de origen y la señal de reloj.

la están basadas en la fotografía. La tecnología de fabricación ha ido evolucionando muy rápidamente, salvando los problemas ocasionados por la necesidad de trabajar con elementos muy pequeños. Para ello se han desarrollado sistemas de diseño asistido por ordenador (CAD, Computer Aided Design) y sistemas de fabricación asistida por ordenador (CAM, Computer Aided Manufacturing), con lo que se han podido abordar problemas de muy difícil solución. De alguna manera se puede hablar de metainformática (informática de la informática): en el diseño de nuevos componentes para ordenadores se utilizan los propios ordenadores.

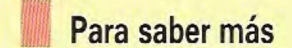
Todos estos avances, tanto de la tecnología de fabricación como del diseño, no han reducido la gran importancia de la labor humana en la producción de nuevos circuitos. Para diseñar totalmente un circuito lógico son necesarios cuatro pasos de relativa complejidad técnica:

- Realización del diagrama de estados y de las tablas de transición.
- Asignación de estados.
- Establecimiento de las ecuaciones de aplicación.
- Diseño final del circuito.

Para dominar esta tecnología sería necesario el estudio de varias materias que no se pueden incluir en una obra de carácter básico. No obstante, daremos a continuación unas nociones elementales sobre cada una de las cuatro etapas.

Diagramas de estados y tablas de transición

El primer paso necesario para diseñar un circuito lógico consiste en describir



Para saber más

¿Qué nivel de tensión se utiliza para cada uno de los estados lógicos?

Ninguno en concreto; es suficiente con utilizar dos voltajes que permitan diferenciar claramente los dos estados lógicos. Sin embargo, lo más usual es utilizar 0 voltios para representar el estado lógico bajo.

¿Cuántas zonas tiene una tabla de verdad?

Dos: la zona de entrada en la que se incluyen todas las posibles entradas y la zona de salida. Esta última contiene los resultados producidos para cada una de las combinaciones de entrada.

¿Cuántas funciones lógicas se pueden representar en un cronograma?

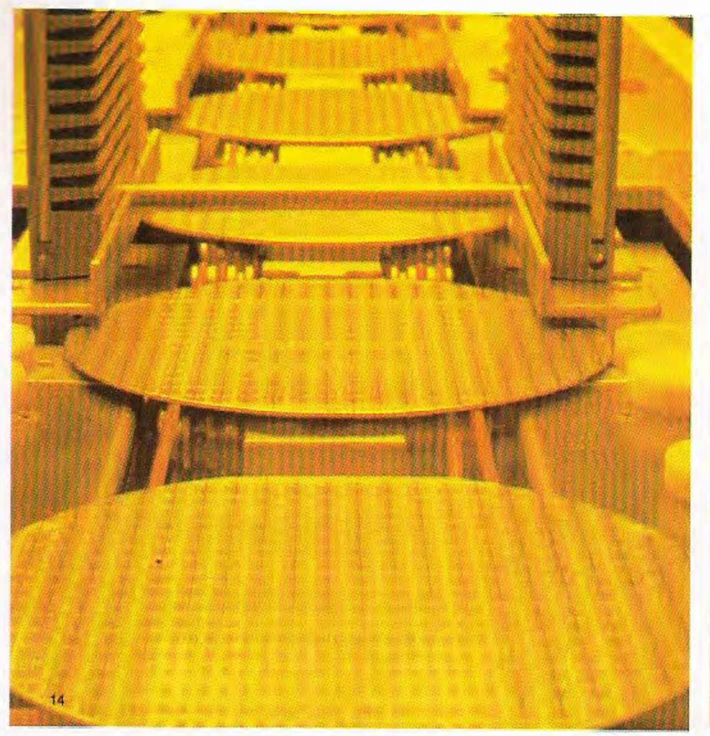
El número de funciones es ilimitado. Normalmente se representan bajo una señal patrón o «reloj», encargada de sincronizar el resto de las funciones.

¿Qué es una puerta lógica?

Es un circuito lógico combinacional con una o más entradas y una sola salida. Las tres operaciones lógicas básicas son: la suma lógica, el producto lógico y la complementación.

¿Cuál es el objetivo de las puertas lógicas: NOR, NAND y OR-EXCLUSIVA?

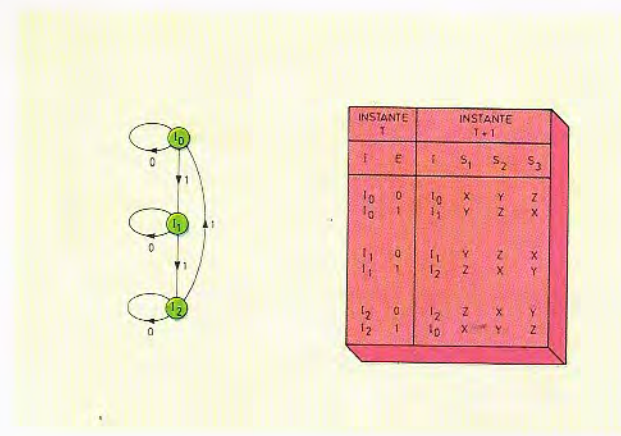
Sintetizar en una misma puerta lógica varias funciones lógicas elementales.



Durante la fase de autograbado, una simple mota de polvo depositada sobre la oblea del material semiconductor, puede llegar a estropear alguno de los circuitos. Esta fase se realiza en unas condiciones ambientales muy cuidadas.



Antes de pasar al proceso de grabación se verifica la distribución de los componentes sobre una reproducción de la máscara ampliada a quinientas o más veces.



INSTANTE T		INSTANTE T+1			
I	E	I	S_1	S_2	S_3
I_0	0	I_0	X	Y	Z
I_0	1	I_1	Y	Z	X
I_1	0	I_1	Y	Z	X
I_1	1	I_2	Z	X	Y
I_2	0	I_2	Z	X	Y
I_2	1	I_0	X	Y	Z

Diagrama de estados y tabla de transiciones de un registro de desplazamiento de tres bits. Para pasar de un estado al siguiente se debe aplicar un uno lógico a la entrada. Si aparece un cero, el estado se conserva.

claramente la función que debe realizar, es decir: cuántas posibles entradas tiene, qué salidas debe producir en cada caso y cuáles serán los estados internos. Se debe realizar, para ello, lo que se denomina diagrama de estados, que consiste en un gráfico con todos los posibles estados internos y la forma en que se pasa de uno a otro.

El diagrama de estados es un dibujo con todos los estados internos representados por círculos. De cada uno de estos círculos, y para cada entrada posible, parte una flecha hacia el estado final al que llega el circuito después de aplicarle una entrada. Sobre dicha flecha se anota también la salida que produce el circuito.

Una vez representado el diagrama de estados es sencillo construir la tabla de transiciones. En ella se pueden distinguir dos partes: en la primera se representan todos los estados internos con todas sus posibles configuraciones de entrada; en la otra se indica el estado resultante de aplicar cada configuración de entrada al estado interno correspondiente y la salida producida. Veamos un ejemplo:

Suponga que queremos diseñar un «registro de desplazamiento de tres posiciones». Tenemos una única entrada, que podrá ser 0 ó 1. Si el estado interno inicial es «X, Y, Z», la salida será «X, Y, Z» para la entrada 0 y «Y, Z, X» para la entrada 1.

Evidentemente, sólo pueden existir tres estados internos: $I_0 \equiv$ «X, Y, Z», $I_1 \equiv$ «Y, Z, X» e $I_2 \equiv$ «Z, X, Y». El diagrama de estados y la tabla de transición correspondientes son los reflejados en la figura.

Asignación de estados

El siguiente paso en el diseño de un circuito secuencial es la determinación del número de biestables necesarios para el funcionamiento del mismo. Este número depende de los posibles estados internos determinados en la etapa anterior.

Continuando con el ejemplo del registro de desplazamiento, en el que tenía-

mos tres estados internos distintos, vemos que son necesarios dos biestables. Cuando ambos contengan el valor 0 se estará en el estado I_0; si el primero vale 0 y el segundo 1, el estado interno será el I_1; si el primero vale 1 y el segundo 0, estaremos en I_2, y en el caso de que ambos valgan 1, corresponde a un estado vacío o inalcanzable.

En general, para representar m posibles estados serán necesarios n biestables; siendo n tal que $2^n \geq m$, y las $2^n - m$ últimas combinaciones representarán estados vacíos.

Una vez asignados los estados a las combinaciones de ceros y unos, podemos construir la tabla de transición completa sin más que sustituir los términos I_i por la configuración que se les haya asociado.

Nº DE CONFIGURACION	ESTADO	CONFIGURACION						
		Q_1	Q_2	Q_3	······	Q_{n-2}	Q_{n-1}	Q_n
1	I_0	0	0	0	······	0	0	0
2	I_1	0	0	0	······	0	0	1
3	I_2	0	0	0	······	0	1	0
4	I_3	0	0	0	······	0	1	1
⋮	⋮	⋮	⋮	⋮		⋮	⋮	⋮
m	I_{m-1}	0	1	1	······	1	1	0
m+1	X	0	1	1	······	1	1	1
⋮	⋮	⋮	⋮	⋮		⋮	⋮	⋮
2^n	X	1	1	1	······	1	1	1

Con «n» biestables tenemos dos configuraciones posibles. En la fase de asignación de estados se determina a cuál de estas configuraciones corresponde cada uno de los estados del circuito que se quiere diseñar.

Ecuaciones de la aplicación

Para cada uno de los elementos Q_i de la tabla de transición completa se debe calcular su ecuación característica; es decir, la expresión que nos da el valor de Q_i en el instante t + 1, en función de todos los elementos $Q_1, \ldots, Q_n$ en el instante t, y de la entrada E.

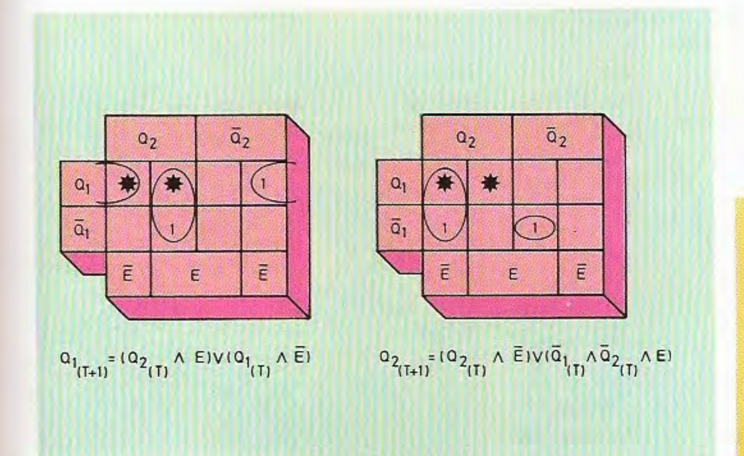
Existen diversos métodos para calcular las ecuaciones que caracterizan a la aplicación; tal vez el más usual sea el de las tablas de Karnaugh. La descripción completa de cualquiera de los métodos es compleja y escapa de los objetivos de esta enciclopedia. En cualquier caso, en la figura se muestra, a título de ejemplo, el cálculo de las ecuaciones de la aplicación para el diseño de un «registro de desplazamiento de tres posiciones».

INSTANTE T			INSTANTE T+1	
Q_1	Q_2	E	Q_1	Q_2
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	1	0
1	0	1	0	0
1	1	0	1✸	1✸
1	1	1	1✸	0✸

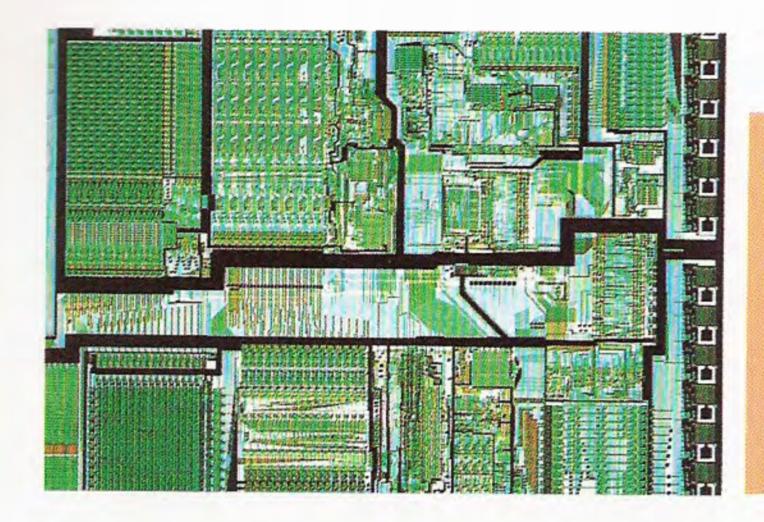
Una vez obtenidas las ecuaciones de aplicación del circuito hay que comprobarlas construyendo su tabla de transiciones. En este caso son correctas, pues la tabla coincide con la que se quería implementar.

Diseño final del circuito

El último paso consiste en la elección de los biestables necesarios según las ecuaciones de la aplicación. Esta etapa del diseño es la menos matemática: las técnicas de diseño, a la hora de conseguir un circuito óptimo, se apoya, sobre todo, en la experiencia del diseñador.



Obtención de las ecuaciones de funcionamiento del mismo registro de desplazamiento, mediante el empleo de las tablas Karnaugh.



Circuito completo, antes de la fase de encapsulado, de un microprocesador de 16 bits. Su tamaño real es de algunos milímetros cuadrados en los que se alojan varios miles de transistores.

	INSTANTE T			INSTANTE T + 1				
	I		E	I		S_1	S_2	S_3
	Q_1	Q_2		Q_1	Q_2			
I_0	0	0	0	0	0	X	Y	Z
I_0	0	0	1	0	1	Y	Z	X
I_1	0	1	0	0	1	Y	Z	X
I_1	0	1	1	1	0	Z	X	Y
I_2	1	0	0	1	0	Z	X	Y
I_2	1	0	1	0	0	X	Y	Z
ESTADO INUTIL	1	1	0	✸	✸	✸	✸	✸
ESTADO INUTIL	1	1	1	✸	✸	✸	✸	✸

Matriz de transiciones completa para el registro de desplazamiento de tres bits. En la parte de la izquierda de la tabla aparecen todas las posibles configuraciones de estados y entradas; en la parte de la derecha se muestran las salidas del circuito y el estado a que llega en cada uno de los casos.

Para saber más

En qué consiste la fase de diseño denominada «Diagramas de estado y tablas de transición?

Es una primera toma de contacto con la tarea que realizará el circuito. Su objetivo final es representar gráficamente, y en forma de tabla, el funcionamiento del circuito.

¿Qué función se realiza durante la fase de «asignación de estados»?

En esta etapa del diseño de un circuito lógico se determina el número de biestables que serán utilizados y se asignan los estados necesarios, según la tabla de transiciones.

¿Qué son las ecuaciones de la aplicación?

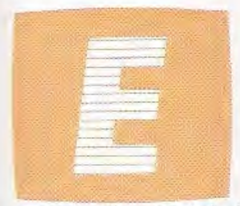
Son funciones booleanas de dos tipos de variables: los estados internos y las entradas al circuito. Con ellas se representan todos los estados de un instante t + 1, en función de los estados y de la entrada en el instante t.

¿En qué consiste la última etapa del diseño de un circuito?

En ella se eligen los biestables que se utilizarán y se determinan las ecuaciones de entrada y salida del circuito.

Circuitos integrados

El nacimiento de un chip



El objeto fundamental de la fabricación de circuitos integrados a gran escala es reducir su tamaño y minimizar los costes. Los patrones de los circuitos integrados se realizan, primero, a gran escala y, posteriormente, se graban fotográficamente en reducidas dimensiones.

Factores de reducción de costes

En los últimos cinco o seis años los procesos para la fabricación de un circuito integrado a gran escala no han sufrido excesivas alteraciones, si bien los costes de fabricación sí han ido reduciéndose progresiva y notablemente.

La explicación es muy sencilla: los circuitos se realizan sobre obleas de silicio y, aunque el precio de éstas se ha visto afectado por la inflación, la superficie necesaria para contener un determinado circuito ha ido disminuyendo constantemente. De esta forma los costes han bajado de forma espectacular. Esta disminución del tamaño se debe a la progresiva miniaturización de los elementos de los circuitos y de las interconexiones necesarias.

Otro de los motivos de la reducción de los costes es la disminución del número de errores cometidos en el proceso de fabricación. Un defecto tan leve como el producido por un pequeño arañazo de algunas milésimas de milímetro significa la inutilidad del circuito.

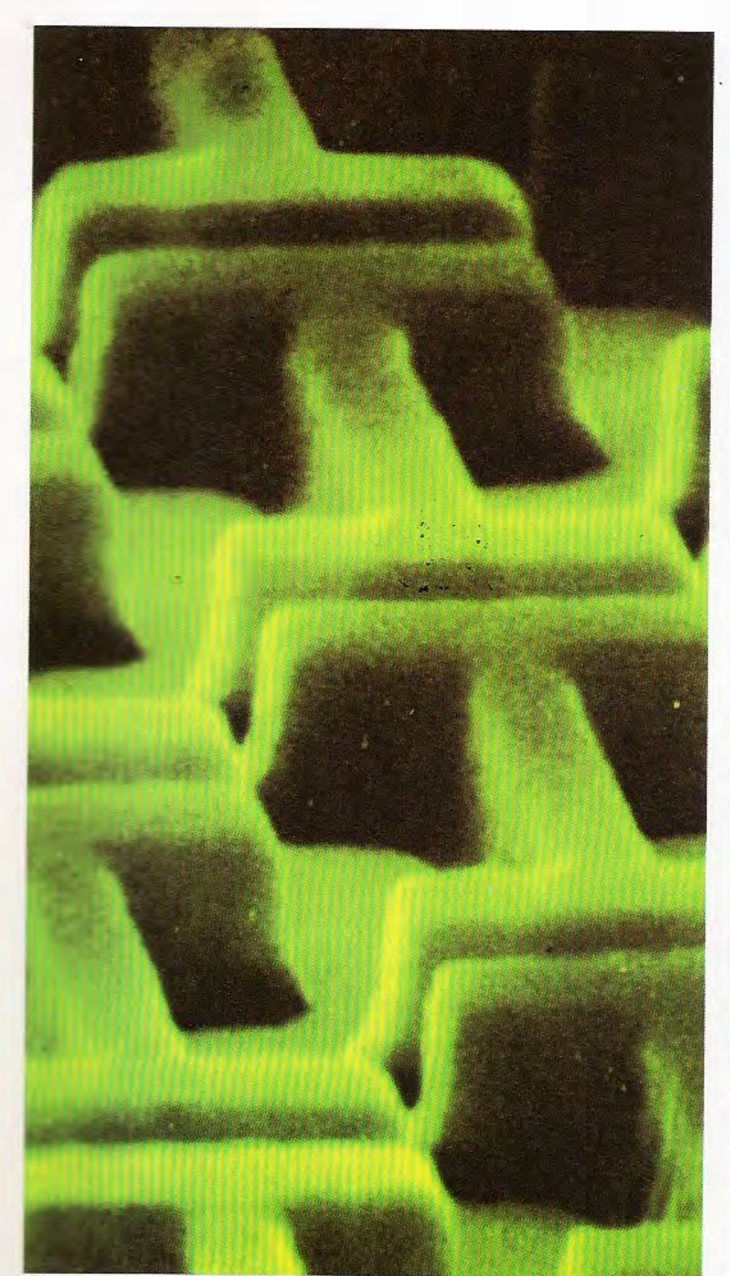
Debido a los mínimos tamaños con que se trabaja, la corrección de los errores resulta más costosa que el proceso completo de fabricación, y, por tanto, si un circuito es defectuoso, se desecha definitivamente.

A pesar de que un circuito integrado a gran escala contiene varios miles de elementos, éstos tienen un tamaño tan reducido que el circuito completo no ocupa más que un cuadrado de 5 a 7 milímetros de lado. Sin embargo, el diámetro usual de las obleas de silicio oscila de 8 a 10 centímetros. De esta forma en la misma oblea se producen simultáneamente cientos de microcircuitos, con la consiguiente reducción de costes. Cuando la oblea ha pasado por todos los pasos necesarios se procede a seccionar las distintas pastillas que constituyen, cada una, un circuito completo.

El objeto fundamental de la fabricación de circuitos integrados a gran escala es reducir su tamaño y, en consecuencia, minimizar los costes y el consumo energético de los circuitos.

Tecnología de fabricación de máscaras

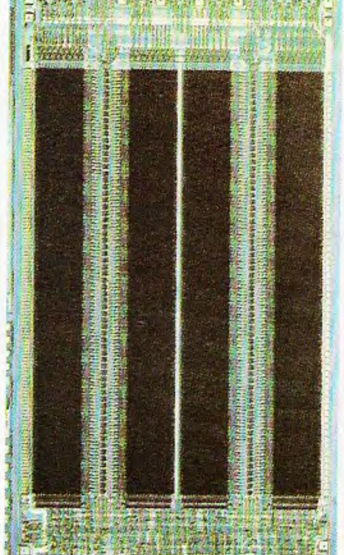
La composición de un circuito integrado es bastante compleja. Para su fabricación se trabaja en tres dimensiones, mediante distintas capas. Cada una de ellas tiene un diseño distinto, al que llamaremos máscara. Algunas de las capas están en el interior de la oblea de silicio y otras en su superficie.

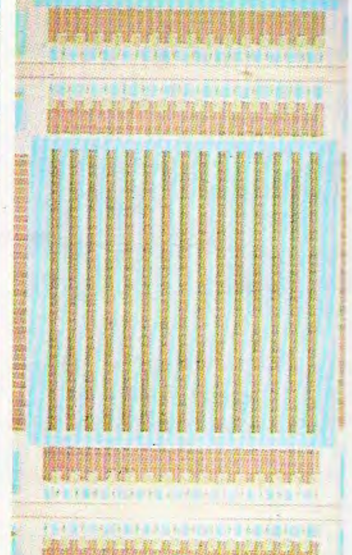


Microfotografía de un circuito de células resistivas creadas sobre la superficie de silicio de un circuito integrado.



Obleas en proceso de tratamiento (sometidas a una atmósfera de plasma) para la obtención de varios cientos de circuitos integrados.





Microfotografía de una memoria RAM estática de 64 Kbits de tecnología CMOS, con varias decenas de miles de transistores integrados. A la derecha aparece una red de puertas lógicas integradas CMOS.

El proceso de fabricación consiste en realizar la superposición de las capas, siguiendo el orden indicado por el diseñador.

Partiendo de las especificaciones funcionales del circuito, y de los procesos necesarios para su fabricación, se realiza el diseño real del dispositivo, calculando el tamaño y la posición que ocuparán los distintos elementos. Para ello se suele utilizar una pantalla gráfica conectada a un ordenador, en la que se representa el circuito ampliado cientos de veces. El producto final de este proceso son las máscaras de cada una de las capas del circuito integrado.

La fabricación de las máscaras se realiza generando un patrón aumentado en el ordenador, que pasará a una placa fotográfica mediante un barrido con un haz luminoso.

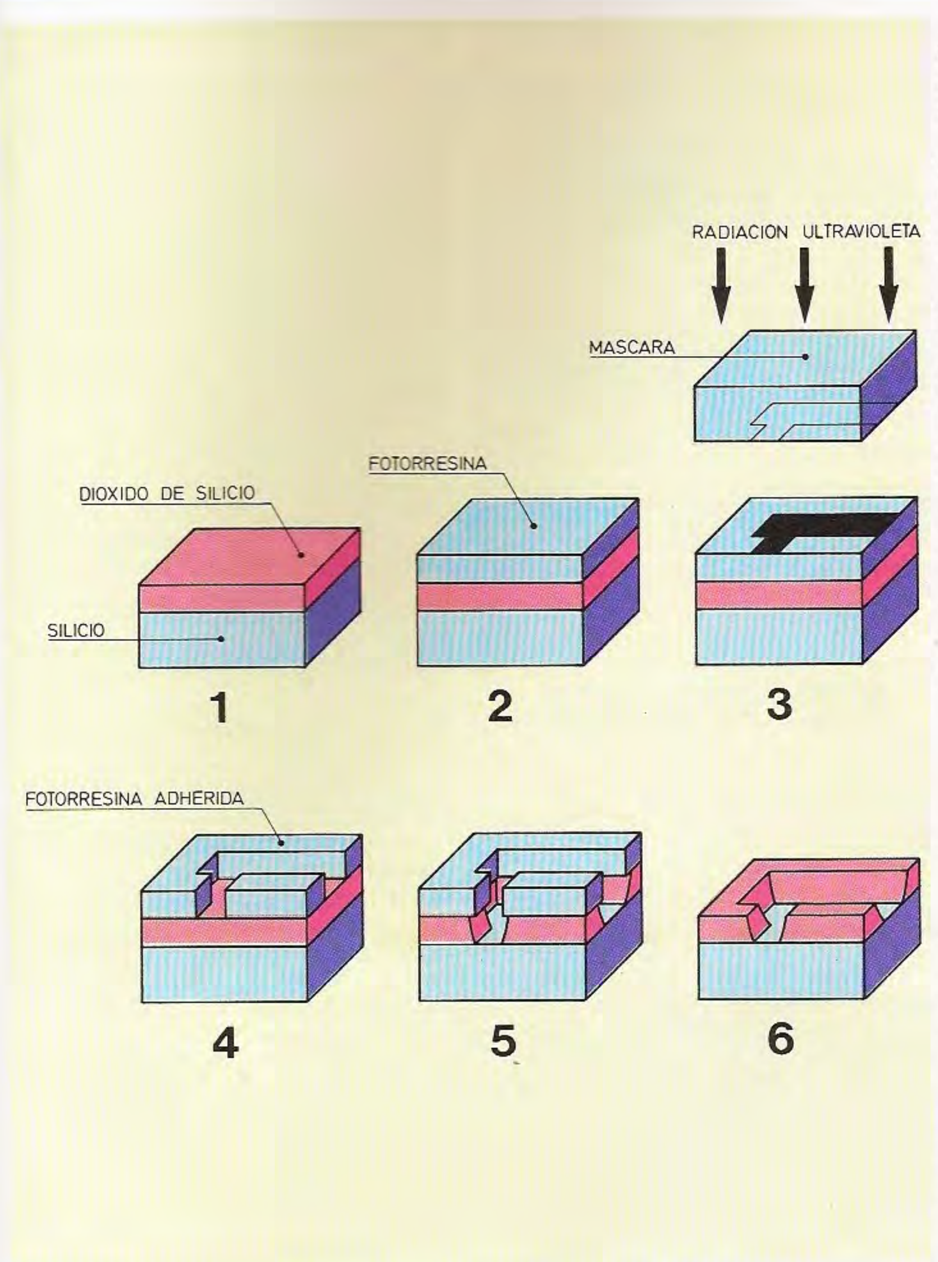


Ilustración gráfica de las seis etapas de que consta un proceso fotolitográfico: 1. Oblea oxidada. 2. Recubrimiento con fotorresina. 3. Exposición a la luz ultravioleta. 4. Atacado del dióxido de silicio con una solución de ácido fluorhídrico. 5. Retirado del patrón de fotorresina. 6. Resultado del tratamiento químico final.

Una vez comprobado este patrón, se proyecta ópticamente sobre la máscara final. Por contacto directo se copian cientos de veces las máscaras y se envían a los centros de fabricación, que utilizan, además de las máscaras, obleas de silicio y productos químicos.

Fotolitografía

La fotolitografía es un proceso de fotograbación que permitirá pasar de las máscaras de las distintas capas del circuito al circuito integrado propiamente dicho.

Sobre una oblea de silicio con una capa superior de dióxido de silicio se coloca una gota de fotorresina diluida mediante un disolvente.

A continuación se hace girar rápidamente a la oblea, con lo que se produce una delgada película fotosensible sobre su superficie; el disolvente se evapora y tan sòlo permanece la fotorresina.

Seguidamente se la expone a la luz ultravioleta a través de la máscara, con lo que la fotorresina se hace insoluble a una solución reveladora, y así se puede retirar de las partes en que la máscara era opaca. Después se sumerge la oblea en una solución de ácido fluorhídrico que ataca selectivamente al dióxido de silicio, manteniéndose el diseño de la fotorresina y el substrato de silicio inalterados.

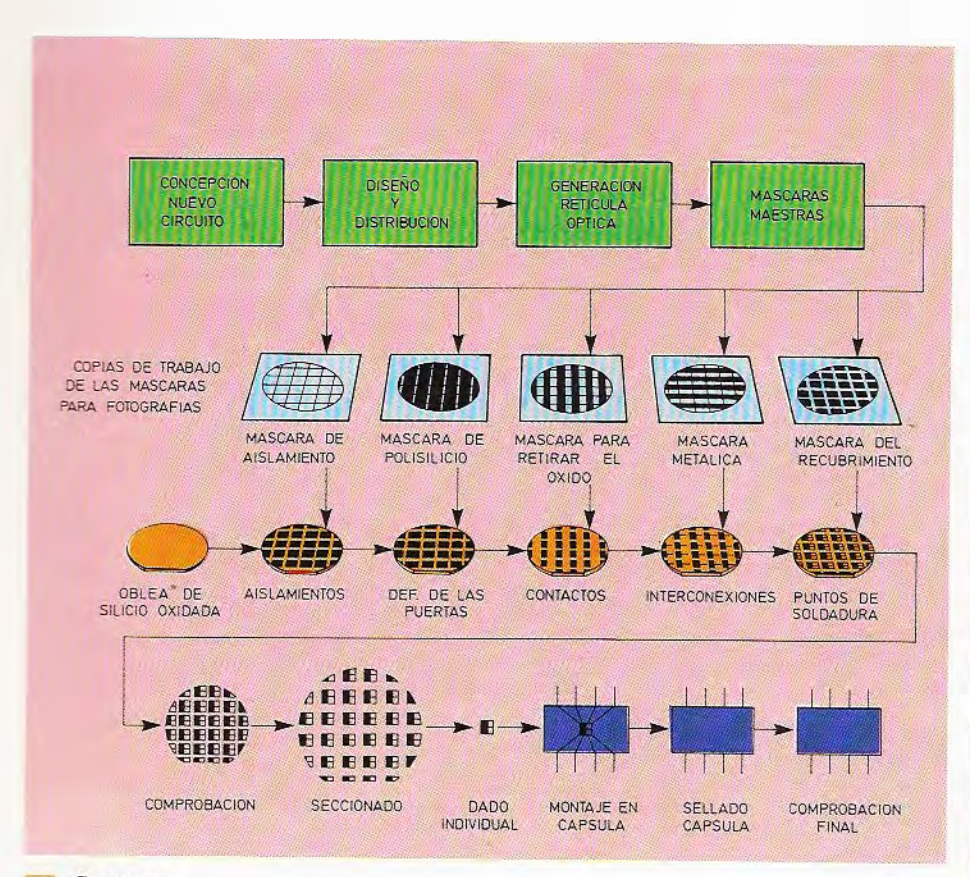
Por último, mediante otro tratamiento químico, se quitan los restos de fotorresina, con lo que la capa queda terminada.

Confección final del circuito

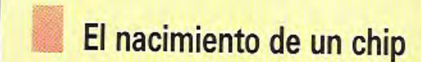
Una vez que están grabados en la oblea todos los circuitos, se procede a comprobar el proceso de fotolitografía. Si ha sido correcto se secciona la oblea de forma que cada uno de los circuitos integrados queden separados en dados individuales. La última operación por la que pasa el circuito es el encapsulado: éste se introduce en una caja protectora que le permita tener interconexiones con otros circuitos. Las cápsulas suelen tener un tamaño muy superior al del circuito.

Una vez soldados los circuitos en las cápsulas, se conectan mediante finos hilos a los electrodos que salen fuera de la cápsula; a continuación la cápsula se sella a fuego, con lo que el dispositivo queda listo para la comprobación final.

Es importante retrasar lo más posible la partición de la oblea en dados individuales, ya que a partir de ese momento los costes de manipulación aumentan enormemente, tanto por la reducción del tamaño de los elementos con que se trabaja como por el hecho de realizar los procesos individualizados y no en grupo.



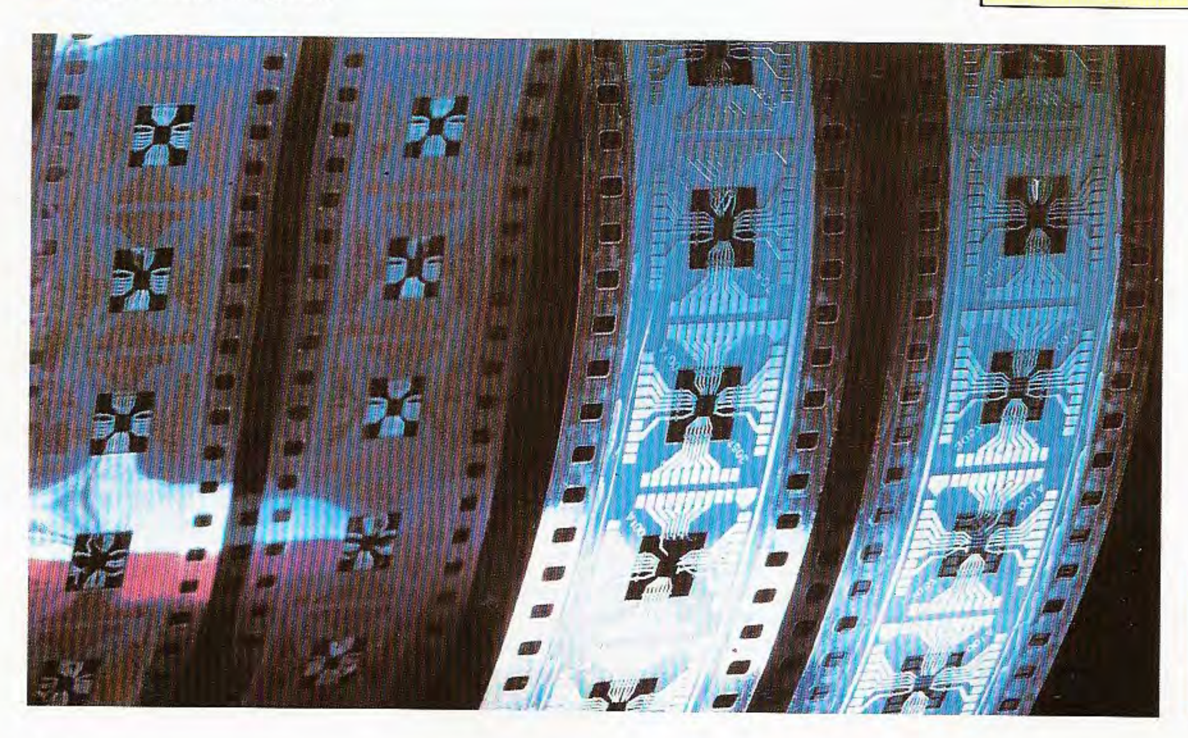
Resumen de las etapas necesarias para la fabricación de un circuito integrado.



El nacimiento de un chip

La fabricación de circuitos integrados a gran escala, detallada en la figura adjunta, consta de las siguientes etapas esenciales:

1. Concepción del circuito integrado detallando los elementos que contendrá y los procesos necesarios para su fabricación.
2. Diseño y distribución del circuito utilizando para ello un ordenador. Se consigue con ello trabajar rápidamente y con mucha precisión.
3. Generación de la retícula óptica, que se compone de las máscaras necesarias para cada una de las capas del circuito.
4. Producción de las máscaras maestras que se utilizan en el proceso de fabricación y copias de los circuitos.
5. Grabación en la oblea original de silicio oxidada de las distintas capas del circuito, a partir de las máscaras (cinco en el ejemplo de la figura).
6. Comprobación de la oblea para determinar los circuitos válidos.
7. División de la oblea en dados individuales que contengan un único circuito.
8. Encapsulado y soldadura de los hilos conectores, y posterior sellado.
9. Comprobación final que garantice el buen funcionamiento del producto.



Circuitos integrados realizados sobre soporte flexible (cortesía Bull).

¿Cómo elegir ordenador?



Cómo hay que elegir un ordenador? ¿Cuáles son los datos a evaluar antes de la elección? ¿Cómo hay que interpretar estos datos y características?... Estas son algunas de las preguntas para las que hay que encontrar respuesta a la hora de adquirir un equipo informático.

A lo largo de los dos próximos tomos de esta obra se exponen las características y peculiaridades de los ordenadores personales —domésticos y de uso profesional— de mayor vigencia y difusión. Para poder evaluar las diferencias y similitudes que existen entre los diversos sistemas es preciso aplicar a todos ellos un cuestionario análogo que evidencie claramente la potencia y posibilidades de cada ordenador.

El avance de la informática y su definitiva penetración en la sociedad actual se apoya sobre la base de los microordenadores. La amplia variedad y diversificación de este tipo de sistemas hacen que su campo de aplicación sea casi ilimitado.

Dentro de los microordenadores que se van a analizar se encuentran sistemas destinados a aplicaciones domésticas, a la resolución de cálculos científicos, orientados a profesionales, a la gestión de empresas... El inventario va a ser exhaustivo y definitorio de las posibilidades que ofrece cada sistema: al usuario corresponde la tarea de conjuntar los parámetros oportunos y elegir el sistema óptimo para satisfacer sus necesidades.

Los microordenadores de los ochenta

Para facilitar la interpretación de los datos que se van a aportar es preciso dar a conocer previamente cuáles son los elementos básicos que intervienen en un sistema ordenador personal y los parámetros que los definen.

La unidad central

El microprocesador

El lugar en el que reside la «electrónica inteligente» del sistema, y que hace que éste sea más o menos potente, se denomina «unidad central».

El «cerebro» de todo microordenador y, en definitiva, de su unidad central es el microprocesador: un circuito integrado que contiene en su interior un chip de

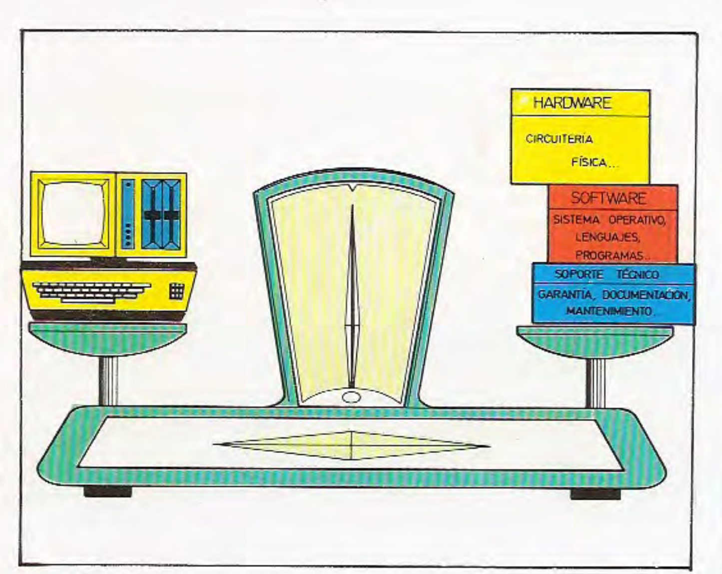
A la hora de evaluar la potencia y posibilidades de un microordenador, hay que observar detenidamente las características del sistema conjunto, no sólo de su unidad central, aunque ésta constituya realmente su columna vertebral.



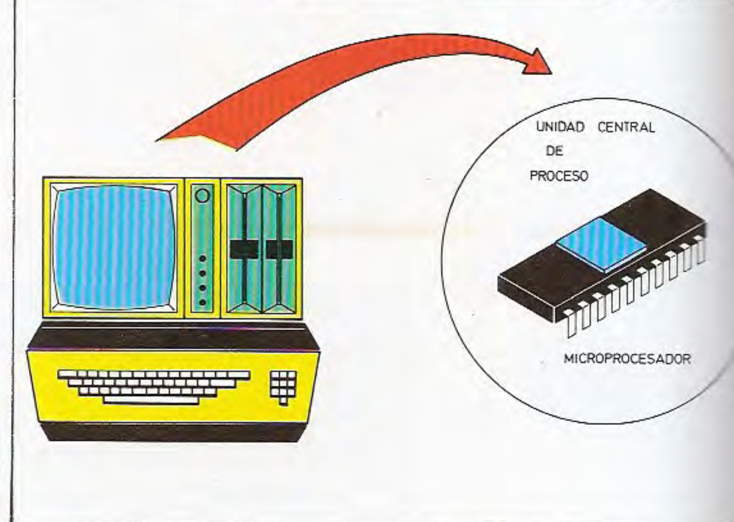
Dentro de la categoría de los microordenadores caben sistemas de muy diversa arquitectura y características. Algunos incorporan en un mismo soporte a la unidad central, el teclado, el órgano de visualización, la memoria de masa e incluso la impresora...



...Por el contrario, otros microordenadores incorporan las diversas unidades constitutivas del sistema conjunto en soportes independientes.



Al proceder al análisis de las características de un ordenador no sólo hay que atender a su estructura circuital; el fiel de la balanza también pueden inclinarlo las posibilidades de programación y el soporte técnico y documental del sistema.



El microprocesador, que constituye el cerebro activo de los microordenadores, determina en gran medida la potencia y capacidad operativa del sistema conjunto.

silicio, no más grande que una uña, y que controla todo el funcionamiento del sistema.

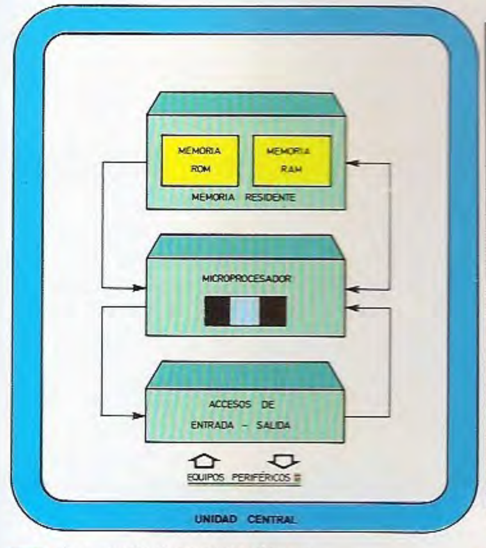
Las características del microprocesador determinan, en gran medida, la potencia y capacidad operativa que va a tener la máquina.

En cuanto a su arquitectura, existen dos grandes familias de microprocesadores: de 8 y 16 bits, aunque ya están al alcance de la mano los primeros equipos personales basados en una arquitectura de 32 bits.

Memoria residente

La memoria residente es la zona de la unidad central destinada al almacenamiento de información; normalmente memoriza programas, más o menos complejos, para su ejecución inmediata.

Si la cantidad de memoria disponible es reducida, el microprocesador podrá acceder a pocos datos directamente, lo cual constituye un factor restrictivo. Los microordenadores vienen de fábrica con un determinado volumen de memoria



La unidad central de un microordenador la constituyen: el microprocesador, la memoria residente y los accesos para la entrada y salida de información.



Unidad central de un microordenador para aplicaciones de gestión. En la zona derecha se observa la incorporación de una unidad de memoria de masa, concretamente un lector/grabador de discos flexibles.



El teclado y la pantalla constituyen los dispositivos periféricos más comunes a través de los que se establece la comunicación entre el ordenador y el usuario o mundo exterior.



La mayor parte de los microordenadores recurren a unidades de disco flexible de 5 y 1/4 ó de 3,5 pulgadas como memorias de masa.

que, normalmente, se mide en Kilobytes. Esta memoria suele ser ampliable, por medio de módulos conectables a la unidad principal, hasta un límite máximo que es característico de cada ordenador.

La zona básica de la memoria principal o residente es del tipo denominado RAM, esto es: memoria de acceso aleatorio (Random Access Memory). La característica de «acceso aleatorio» indica, en resumidas cuentas, que el «cerebro de la máquina» (el microprocesador) puede grabar (escribir) o extraer (leer) datos en la memoria sin tener que seguir ningún orden particular.

Otro tipo de memoria incluida en la unidad central es la que obedece al apelativo de ROM o de memoria de sólo lectura. En ella el fabricante graba las instrucciones básicas que coordinarán el funcionamiento del microordenador. Las instrucciones que tiene grabadas la ROM suelen ser exclusivas de cada máquina e incluso defendidas por un «copyright».

Líneas de comunicación

La tercera zona básica de la unidad central de cualquier ordenador es la constituida por los elementos que permiten la transferencia de información en ambos sentidos: entrada y salida.

Estos elementos de acceso suelen adoptar la denominación anglosajona de «ports»: puertos para la entrada o salida de unas mercancías muy especiales, los bits (ceros y unos). A ellos se conectan los dispositivos que aportan o reciben información «al o desde» el microordenador.

Básicamente, existen dos tipos de accesos o «ports»: serie o paralelo. Esta distinción se refiere únicamente al formato en el que se transfieren los datos: si es un bit a la vez el que se transmite, el acceso será de tipo serie, mientras que si son varios los bits que se canalizan simultáneamente, el tipo de acceso o comunicación será paralelo.

Uno de los accesos («ports») más importantes es el que se utiliza para conectar la impresora al ordenador, y uno de los formatos más comunes es el de tipo serie que responde a las siglas RS/232.



A la hora de elegir impresora hay que tener en cuenta, en primera instancia, que su formato de comunicación (serie RS/232, paralelo «Centronics»...) coincida con el que incorpora el microordenador.



El campo de aplicación de un microordenador está ligado con sus posibilidades de programación que dependen de los sistemas operativos y lenguajes disponibles, de los programas y del soporte técnico que proporcione el fabricante.

El teclado

El teclado es el órgano más común a través del que el usuario se comunica con el ordenador, introduciendo la información oportuna. En algunos casos, el teclado está integrado en la misma caja que contiene a la unidad central; en otros está separado físicamente de la misma, manteniendo la unión a través de un cable.

El número de teclas tiene su importancia en cuanto a que permite una mayor o menor flexibilidad en el manejo de la máquina.

Al hablar de teclados podemos referirnos a dos tipos básicos: el tipo más profusamente utilizado es el denominado QWERTY; dicha palabra corresponde al orden en el que están dispuestas las teclas de la primera fila. Existe una segunda variante, que obedece a las siglas AZERTY.

Las diferencias entre ambos tipos básicos de teclado no tienen mayor importancia a la hora del funcionamiento del microordenador.

Por otra parte, las teclas pueden estar agrupadas en un bloque único —de forma semejante a una máquina de escribir— o distribuidas en varias zonas. Cada vez son más los ordenadores que disponen de un teclado numérico separado; éste resulta de gran utilidad cuando hay que introducir frecuentemente una gran cantidad de datos numéricos.

También es frecuente que el teclado disponga de un bloque independiente de teclas «funcionales», teclas que al accionarlas ejecutan directamente o escriben en el programa un conjunto de instrucciones más o menos complejas.

La pantalla

El tercer elemento que entra en juego es la pantalla: el dispositivo más frecuentemente utilizado por los ordenadores para presentar sus datos y respuestas al usuario.

En este punto existen dos posibilidades esenciales: los microordenadores domésticos, que suelen utilizar un televisor doméstico en color o blanco y negro, y los equipos de uso profesional, que utilizan un monitor (también en blanco y negro o color), aunque en este último caso es frecuente que la pantalla sea de fósforo de color verde o ámbar.

Una característica importante de la pantalla es su formato, definido por el número máximo de filas y columnas de caracteres que pueden aparecer simultáneamente sobre la misma.

Otra propiedad importante es la posibilidad de obtener gráficos sobre la pantalla y si éstos son de alta o baja resolución. En el caso de las pantallas de color es oportuno conocer el número de colores generables y en las de blanco y negro, su escala de grises... Y en cualquier

caso, es preciso conocer su resolución en número de «pixels» (elementos de imagen) horizontales y verticales.

Memorias de masa

Cuando es necesario almacenar una gran cantidad de información: programas o datos, se recurre a las llamadas memorias de masa o, lo que es lo mismo, a dispositivos de alta capacidad que memorizan indefinidamente sobre un determinado soporte todos los bits y bytes que deseemos almacenar.

En la actualidad, las memorias de masa más utilizadas con los microordenadores son los discos flexibles (disquetes) y los discos rígidos, también denominados discos duros.

Los disquetes o «floppy disks» son discos magnéticos flexibles que vienen protegidos dentro de una funda de papel y cuya capacidad de almacenamiento varía entre límites muy diversos. En lo relativo a tamaño, los discos flexibles más corrientes son los de 5,25 pulgadas y 3,5 pulgadas.

El principio de funcionamiento de los discos rígidos es semejante al de los flexibles, aun cuando existe una diferencia sustancial en su tecnología y capacidad de almacenamiento.

Dispositivos periféricos

El concepto informático asociado a este término coincide con su significado más literal. Los periféricos son los dispositivos a través de los que el ordenador se comunica con el exterior, captando información o entregándola. Dentro de este grupo están las impresoras, los módems para comunicación telefónica entre ordenadores, los plotters o trazadores gráficos e incluso las palancas (joysticks) para juegos.

Sistemas operativos y lenguajes de programación

A la hora de analizar un ordenador no sólo hay que tener en cuenta sus elementos físicos o circuitería (hardware); también hay que evaluar sus posibilidades y herramientas de programación (características software).



Unidad central —con doble unidad de disco flexible integrada— y teclado del ordenador personal Toshiba T1500, compatible IBM-PC.

En esta etapa hay que tener en cuenta cuál es ei sistema operativo que incorpora o, lo que es lo mismo, el conjunto de programas cuya misión consiste en coordinar la actuación global del ordenador.

El sistema operativo (denominado programa monitor en el caso de pequeños ordenadores) puede estar almacenado dentro de una o varias unidades de memoria de sólo lectura (ROM) o, habitualmente, grabado en un disco flexible.

Además del sistema operativo, el ordenador precisa de algún programa traductor que le permita interpretar y, en consecuencia, ejecutar las órdenes del usuario. El traductor en cuestión debe ser «experto» en el lenguaje de diálogo elegido. Este lenguaje suele ser de los denominados de «alto nivel»: por ejemplo, BASIC, FORTRAN, PASCAL, COBOL, LOGO...

En definitiva, los lenguajes de alto nivel empleados por los ordenadores constan de un determinado vocabulario (normalmente reducido) de palabras que sirven para formular las instrucciones, que a su vez constituyen los programas. El lenguaje más utilizado en los microordenadores actuales es el BASIC.



Una de las peculiaridades del Hewlett-Packard 150 II reside en el accesorio de pantalla táctil, el cual permite controlar las aplicaciones sin más que apoyar el dedo sobre la zona adecuada de la pantalla.

Y, por supuesto, no hay que olvidar un examen a la biblioteca de programas de aplicación. Su importancia es tal que, realmente, son los responsables de que el amasijo de metal y plástico denominado ordenador se convierta en una herramiento útil, eficaz e irrenunciable en muchos casos.

Indice temático

Tratamiento de la información

Algoritmos y programas

Análisis de sistemas

Análisis de problemas técnicos y de gestión

Almacenamiento organizado de la información

Bases de datos

Bases de datos para microordenadores

Procesadores de texto

Hojas electrónicas

Su majestad el microprocesador

La información en el microprocesador

Registros de los microprocesadores

Métodos de direccionamiento

Microprocesadores de 8 y 16 bits

Circuitos lógicos

Circuitos integrados

¿Cómo elegir ordenador?

